# Contribution à l'étude de la conduite des systèmes de désassemblage.

Luminata Duta

N° ordre : 1162

UNIVERSITÉ FRANCHE-COMTE DU BESANCON

en association avec

UNIVERSITÉ POLYTECHNIQUE DU BUCAREST

# CONTRIBUTION A L'ETUDE DE LA CONDUITE DES SYSTEMES DE DESASSEMBLAGE

**THÈSE**

présentée à
BUCAREST le 22 septembre 2006

pour obtenir le

GRADE DE DOCTEUR DE L'UNIVERSITE DE BUCAREST
GRADE DE DOCTEUR DE L'UNIVERSITE DE FRANCHE-COMTE

en Automatique et Informatique

PAR

**Luminita DUTA**

Soutenue devant le Jury :

| | | |
|---|---|---|
| Président : | D. POPESCU | Professeur, Université Polytechnique de Bucarest |
| Directeurs de thèse : | | |
| | J.M. HENRIOUD | Professeur, Université de Franche-Comté de Besançon |
| | Fl. Gh. FILIP | Viceprésident de l'Academie Roumaine |
| Rapporteurs : | Al. DOLGUI | Professeur, Ecole Nationale Supérieure de Mines de Saint Etiènne |
| | V. MINZU | Professeur, Université "Dunarea de Jos" de Galati |
| Examinateur : | N. ZERHOUNI | Professeur, Ecole Nationale Supérieure de Mécanique et Mécatronique de Besançon |
| Invité : | I. CUCUI | Recteur de l'Université Valahia Targoviste |

***A mes parents,***

# REMERCIEMENTS

Ce travail de recherche a été rendu possible grâce au support financier Français et a été réalisé principalement au Laboratoire d'Automatique de Besançon, France. Je remercie M. Alain BOURJAULT, le directeur du LAB, pour les conditions de travail exceptionnelles qu'il a mis à ma disposition.

Je veux exprimer ma profonde reconnaissance aux deux co-directeurs de thèse : M. Jean Michel HENRIOUD, Professeur à l'Université Franche-Comté et M. Florin Gh. FILIP, vice-président de l'Académie Roumaine de Sciences, qui m'ont guidé avec patience, exigence et professionnalisme dans l'accomplissement de ce travail.

Je remercie M. Alexandre DOLGUI, Professeur à l'Ecole des Mines et Mécanique de Saint–Etienne et à M. Viorel MINZU, professeur et vice-président de l'Université Dunarea de Jos de Galati, qui m'ont fait l'honneur d'être les rapporteurs de cette thèse.

Mes remerciements à M. Ion CUCUI, le recteur de l'Université Valahia de Targoviste, qui m'a encouragé en permanence et m'a assuré le support financier nécessaire pour les déplacements à étranger.

Je ne peux oublier M. Nicolae OLARIU, Professeur à l'Université Valahia de Targoviste qui a eu confiance dans ma capacité de travail et qui m'a encouragé à faire cette thèse en cotutelle et M. Nourredine ZERHOUNI, Professeur à l'ENSMM Besançon qui a soutenu ce projet.

Mes remerciements singuliers à ma collègue Mme. Eugénia MINCA, Maître de Conférences à l'Université Valahia de Targoviste, qui a toujours été à mes côtés pendant mes recherches en France.

Je remercie le personnel du LAB en entier pour la chaleur et la générosité avec lesquels ils m'ont accueilli. Mes remerciements à mes amis du LAB : Frederic, Ivanna, Roberto, Sid-Ali, Ryad, Laurent, Karima, Alex, Paris, Mehdi, Slava, Blaise et Jihene.

Il n'y a pas des mots appropriés pour remercier ma famille : mon mari Marius pour son amour et sa confiance ainsi ma fille Elena pour ses encouragements. Je dédie cette thèse à mes parents Ilinca et Simion pour leur exprimer ma reconnaissance pour tous les efforts qu'ils ont fait en vue de me donner une bonne éducation spirituelle et professionnelle.

Les derniers remerciments et pas des moindres, je le reserve aux médecins neurochirurgiens Exercian et Soare qui ont sauvé ma vie et m'ont donnée ainsi la possibilité de finir ce travail.

# TABLE DES MATIÈRES

# LISTE DES FIGURES

# LISTE DES TABLEAUX

# LISTE DES ABRÉVIATIONS, DES SIGLES ET DES ACRONYMES

| | |
|---|---|
| ADD | Arbre de Décision du Désassemblage |
| AG | Algorithmes Génétiques |
| ALBP | Assembly Line Balancing Problem |
| BC | Base de Connaissences |
| BD | Base de Donées |
| DLBP | Disassembly Line Balancing Problem |
| DPL | Decision Programming Language |
| GPAO | Géstion de la Production Aidée par l'Ordinateur |
| K | Algorithme Kangourou |
| LAB | Laboratoire d'automatique de Besançon |
| ME | Méthode Exacte |
| MRP | Material Requirements Planning |
| MRP II | Manufacturing Resources Planning |
| PBM | Plan des Besoins Matières |
| PDP | Programme Directeur de Production |
| RdP | Reseau de Petri |
| SAD | Système d'Aide à la Décision |
| SCC | Système de Contrôle et Commande |
| SGBD | Système de Gestion des Basses de Données |
| SIAD | Système Informatique d'Aide à la Décision |
| SVA | Système de Vision Artificielle |
| VMA | Valeur Monetaire Attendue |

***Ce qui  ne te tue pas, te rend plus forte...***

*(Nietzsche)*

# INTRODUCTION GENERALE

Le domaine de la productique est relativement nouveau pour les chercheurs de la Roumanie. Les derniers cinq ans ont été favorables au développement de l'industrie roumaine. La modernisation et l'automatisation des entreprises sont à l'origine de ce développement à la suite desquelles les industriels se sont de plus en plus intéressés à la recherche dans le domaine de la productique.

Cette thèse a été possible suite de l'intérêt porté par l'équipe managériale de l'entreprise "Dacia-Renault" du Pitesti, Roumanie. Le parc de véhicules hors d'usage (VHU) est très grand en Roumanie. L'intérêt porté à ce problème de VHU est multiple. D'une part, le projet d'intégration de l'Europe (imminent à la veille de l'année 2007) impose, plus que jamais, à l'industrie Roumaine et aux infrastructures étatiques le respect infaillible de l'environnement en général et aux directives environnementales européennes en particulier. D'autre part, ces industriels doivent pouvoir rendre effectif le désassemblage de ces VHU tout en maximisant le profit obtenu et de minimiser les coûts du processus de production correspondant. De plus, ces mêmes industriels (constructeurs automobiles, équipementiers et sous-traitants de manière générale) espèrent non seulement se conformer à ces directives de la manière la plus profitable … en tout cas la moins coûteuse, mais aussi encourager la consommation dans le parc automobile du neuf ! En effet, cela pourrait inciter les partenaires étatiques et industriels (à l'image des pays de l'Union Européenne) à proposer des primes environnementales et des réductions notables sur les produits neufs dont pourront ainsi bénéficier les Roumains. Un intérêt supplémentaire et non des moindres est de créer de l'emploi à travers des filières de valorisation de VHU et de doper l'exportation des produits valorisé devenu ainsi plus compétitives.

C'est à partir des ces enjeux essentiels que ce travail de recherche s'est initié, s'orientant précisément dans le domaine du désassemblage des produits en fin de vie. Ces recherches ont été effectuées dans le Laboratoire d'Automatique de Besançon (CNRS-UMR n° 6596), France. C'est le fruit d'une convention en cotutelle entre l'Université Franche-Comté et de l'Université Polytechnique de Bucarest. Cette thèse a pour but de fournir une méthodologie et d'initier des outils qui permettraient d'atteinte les objectifs cités ci-dessus et ce en traitant les deux points cardinaux suivant :

1. l'accomplissement de la commande d'une ligne de désassemblage par l'ordonnancement statique et dynamique des produits sur la ligne,
2. l'implémentation informatique d'un outil qui renseignera les opérations d'évaluation et d'optimisation du processus.

Ce travail sera présenté en cinq chapitres. Le premier chapitre est une introduction à la problématique du désassemblage. Il vise à préciser la position des résultats obtenus dans

le cadre de cette recherche et met l'accent sur une classification de la littérature par rapport aux niveaux de recherche. Nous mettrons en évidence le fait que la plupart des travaux dans le domaine du désassemblage sont concentrés autour de la planification des opérations ou de l'élaboration des gammes de désassemblage. Il y a peu de travaux qui concernent les niveaux de l'ordonnancement et de la commande du processus. A l'origine, la difficulté qui réside dans la présence des aléas dans le processus de désassemblage. Les opérations de désassemblage sont soumises aux incertitudes, dues aux défauts rencontrés au produit désassemblé en particulier et tous simplement la méconnaissance *a priori* de l'état du VHU à son arrivé dans le système de désassemblage. Il est ainsi naturel de concevoir un système de pilotage réactif aux évènements imprévus pour faire face à ces aléas.

Le deuxième chapitre représente l'état de l'art des systèmes de pilotage de l'assemblage. Bien que différent du processus de désassemblage, il en deumeure pas moins que les fondements –nous le verrons–, restent les mêmes. Seule subsiste la précaution de considérer les particularités du désassemblage notamment ceux de la commande qui sont soulignées à la fin du chapitre. Les architectures de conduite et les niveaux de la commande de l'assemblage y seront préalablement présentés.

Dans le chapitre trois, la problématique de l'ordonnancement du désassemblage est détaillé. Elle est traitée sous deux aspects : l'affectation des tâches aux postes de manière à obtenir un bon équilibrage de ligne et du point de vue économique en maximisant le profit obtenu à la suite du désassemblage du produit. Les équations de l'ordonnancement sont présentées dans le même chapitre. Deux fonctions objectif sont proposées : une fonction de coût (exprimée par le rapport entre le revenu final et le temps de cycle du désassemblage) et une fonction d'équilibrage (dérivée de la fonction d'équilibrage pour l'assemblage, mais tenant compte des contraintes spécifiques au processus de désassemblage). La définition des équations de l'ordonnancement du désassemblage en temps réel garantie le réalisme du problème et nous assure, accessoirement, l'originalité de l'approche.

Le quatrième chapitre propose trois méthodes d'optimisation pour les fonctions objectives. La première est une méthode classique basée sur l'algorithme *Backtracking* qui donne la valeur optimale mais à une complexité proportionnelle au nombre des composants du produit. En pratique, les industriels désirent un résultat rapide qui n'est pas dépendante du type de produit. En réponse à ce fait, les deux autres méthodes d'optimisation proposées sont empruntées à la classe des méthodes approchées. Même si elles ne donnent pas la solution optimale, s'arrêtant parfois dans un optimum local, elles sont plus rapides que la technique classique qui reste, au pire, un moyen d'apprécier l'efficacité des méthodes approchées. Il s'agit de l'application des algorithmes génétiques et d'un algorithme stochastique (Kangoroo). A la fin du chapitre, les résultats donnés par les trois techniques de programmation sont interprétés et comparés.

Dans le chapitre cinq, une architecture de système de commande est proposée. L'originalité de cette proposition est l'intégration d'un système d'aide à la décision dans l'architecture du système informatique. Le désassemblage comme processus implique des décisions qui concernent le type de désassemblage (propre ou destructif), l'affectation des opérations aux postes de travail, la continuation du désassemblage en

fonction du rapport profit/coût. De même, l'ordonnancement en temps réel nécessite un système interactif d'aide à la décision.

Pour lever tout handicape d'adaptation multi-plateforme et privilégier les développements annexes, les interfaces de commande sont conçues entièrement sous l'environnement Visual C++ 6.0. Elles intègrent une analyse de l'équilibrage de la ligne et un module d'aide à la décision.

L'étude de l'ordonnancement et de la commande du processus de désassemblage n'est qu'au début. A la suite de l'implémentation des résultats de la recherche dans des systèmes réels de désassemblage, les méthodes proposées pouront être analysées et perfectionnées. L'auteur de cette thèse espère continuer ses recherches dans ce domaine du désassemblage des produits en fin de vie et d'appliquer les résultats dans les entreprises de Roumanie.

# CHAPITRE I

# PROBLÉMATIQUE DU DÉSASSEMBLAGE

## 1.1. Contexte

Pendant le siècle dernier, des efforts considérables ont été déployés pour améliorer la production industrielle tant sur le plan qualitatif, que quantitatif. Cela s'est soldé par une exploitation effrénée des ressources naturelles et l'apparition des décharges de produits techniques dont le volume grandit avec la densité des consommateurs, ce qui a entraîné la dégradation de la qualité de l'environnement et le tarissement de certaines ressources non renouvelables.

Le développement des techniques, la diminution des ressources naturelles, l'augmentation de la population, les demandes de produits de plus en plus divers, les déchets résultants du processus de fabrication, la mise en décharge, sont des événements qui sont à la base d'une dégradation importante de l'environnement.

Pour parer à ce problème et réduire la consommation de ressources naturelles et le volume des déchets, il est nécessaire de mettre en œuvre des méthodes efficaces de recyclages des produits en fin de vie.

Ainsi, les concepteurs et les fabricants des nouveaux produits sont obligés de prendre en considération les effets de leurs actions sur l'environnement et de concevoir leurs produits afin de faciliter le recyclage.

### 1.1.1 Le recyclage

Il existe actuellement deux sortes de recyclage des produits techniques, à savoir le recyclage "brut" qui a pour objectif la récupération des matières premières par le biais de la démolition et le recyclage "noble" dont le but est la récupération et la réutilisation de composants des produits usagers.

La première méthode consiste à récupérer des matières premières dont une partie est revalorisée en énergie. Son exécution est facile à mettre en œuvre, mais elle est d'une efficacité qui n'est pas assez satisfaisante du point de vue écologique.

La deuxième méthode consiste à extraire, par le biais du désassemblage à partir des produits en fin de vie, les composants qui sont susceptibles d'être réutilisés. Ainsi ces composants seront mis dans un nouveau cycle de vie, après avoir subi un processus de reconditionnement. Cette méthode est intéressante du point de vue écologique puisqu'elle a pour objet de réduire au minimum la quantité de déchets qui résulterait du recyclage des produits techniques en fin de vie. Par ailleurs, elle pourrait être économiquement profitable si les pouvoirs publics des pays industrialisés mettaient en place des réglementations qui encourageraient la vente des produits issus du recyclage.

### 1.1.2 La législation

La législation écologique actuelle oblige les entreprises à respecter des règles très strictes sur les méthodes de recyclage des produits manufacturés. Aujourd'hui les fabricants conçoivent les produits pour faciliter leur recyclage. Dans le même temps, l'évaluation écologique des options de fin de vie pour les produits usagés doit tenir compte de l'ensemble des impacts environnementaux.

Dans l'Union Européenne les lois sont très strictes en ce qui concerne les déchets ménagers et industriels. Dans des pays comme l'Allemagne, les Pays-Bas et la France les lois sur le retour des produits en fin de vie sont déjà en vigueur dans certains secteurs.

En 1998 la Commission Européenne a introduit une directive en ce qui concerne la production de résidus de produits électriques et électroniques: Waste Electrical and Electronical Equipement (WEEE) qui a été modifiée la dernière fois en 2002 [Diréctive Européene, 2002]. Cette normative implique une fabrication qui prend en compte l'environnement et qui suppose une politique de fabrication qui inclut tout le cycle de la vie du produit; elle favorise le recyclage et la réutilisation de composants ainsi que l'élimination de déchets.

En France le gouvernement s'intéresse à la valorisation des produits manufacturés. Un accord cadre a été signé entre le gouvernement français et les représentants de toutes les industries du secteur automobile, qui prévoit qu'en 2015 le pourcentage valorisé d'un véhicule sera de 90% avant sa mise en décharge.

En Allemagne, la loi qui prévoit la neutralisation des déchets ménagers et la valorisation des matériaux plastiques, du papier, du carton, des emballages en verre est en vigueur.

Les produits durables comme les machines à laver, les réfrigérateurs, les télés, l'équipement du bureau (téléphone, fax, copieur, imprimante, etc.) et les ordinateurs personnels, sont considérés aussi des produits ciblés pour le processus de recyclage.

Cette tendance n'est pas limitée au pays de l'Union Européenne. Depuis la moitié des années '90, le grand consortium USCAR (United States Council for Automotive Research) s'est intéressé à la recherche et au développement dans le domaine de désassemblage et du recyclage des voitures mises en décharge.

### 1.1.3 La récupération

En réponse à la grande quantité de résidus de produits industriels, deux tendances dans le domaine de la conception des produits peuvent être dégagées. La première est de concevoir des produits qui n'affectent pas l'environnement et la deuxième qui utilise des techniques pour récupérer les composants des produits usés et traiter les résidus. Dans ce sens, on peut signaler trois manières de concevoir un produit: conception pour recyclage, conception pour l'environnement et conception pour le désassemblage.

Le désassemblage est le processus qui permet la récupération de certains parties du produit par séparation de ses composants. La phase de réutilisation permet la récupération de certains matériaux et composants ou sous-ensembles, et nécessite une technique de désassemblage non-destructive. Par contre, dans l'étape de recyclage des procédés chimiques peuvent être utilisés pour séparer les matériaux ce qui suppose des techniques plus ou moins destructives dans le processus de démontage. Tous ces concepts sont liés entre eux et leur application donne une solution au problème environnemental.

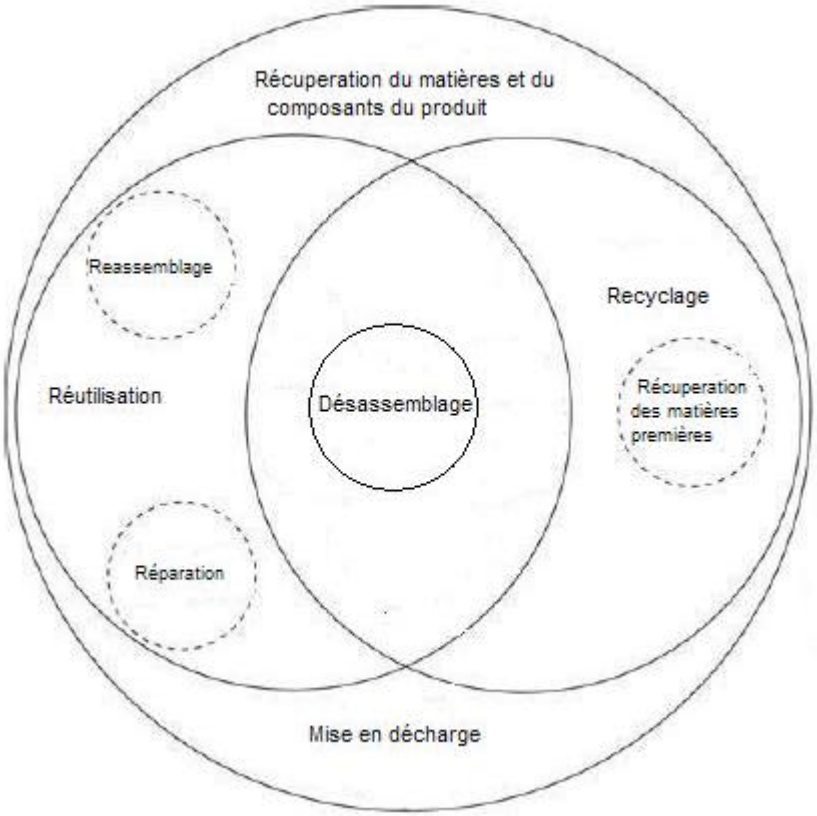



Fig. 1. Le désassemblage à l'intersection du recyclage avec la réutilisation.

Plusieurs recherches sont apportées dans le domaine de conception pour le désassemblage. Touzanne dans sa thèse [Touzanne, 2002] a consacré un chapitre entier pour l'état de l'art de la conception pour le désassemblage.

Ces recherches sont concentrées sur l'intégration de la conception du produit et du désassemblage dans le but de faciliter la revalorisation économique. Trois domaines d'étude sont visés au-delà : l'analyse économique, la génération des gammes de désassemblage et la conception pour le désassemblage.

Notre travail de recherche a pour but l'optimisation du processus de désassemblage du point de vue économique et technique.

### 1.2 Cycle de vie d'un produit

Le cycle de vie d'un produit peut-être défini comme la totalité des étapes parmi lesquelles le produit passe : la fabrication des composants, l'assemblage, le transport, la maintenance, l'utilisation et le recyclage. L'intégration des étapes de récupération et de recyclage est prise en considération dans le cycle de vie d'un produit ce qui fait reconsidérer les notions de qualité, innovation et efficacité dans le processus de fabrication du produit (figure 2).

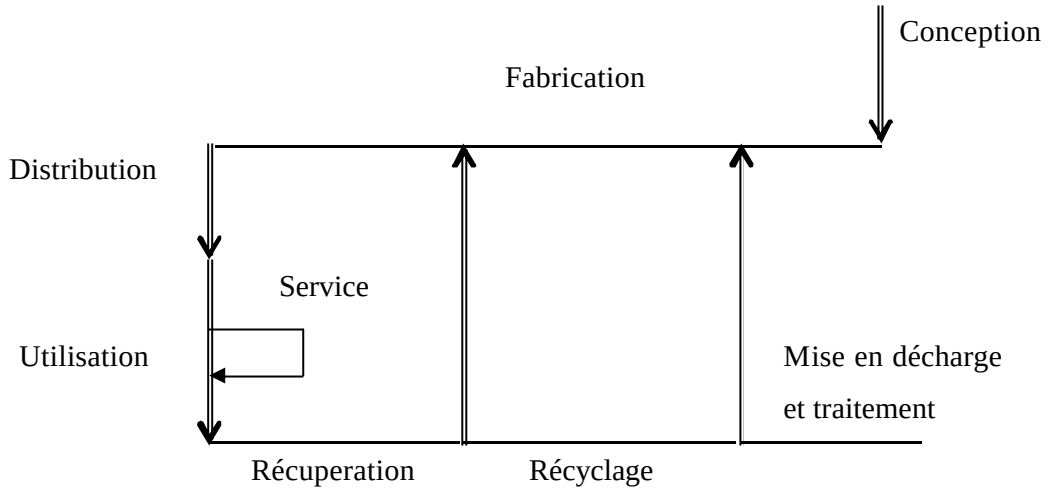



Fig. 2. Le cycle de vie du produit (selon Filip, 1999)

Le processus de recyclage intervient en fin de vie d'un produit. Le but du recyclage est la valorisation. La valorisation est un traitement destiné à redonner une valeur à un sous-ensemble ou à un constituant d'un produit par réemploi, récupération du potentiel énergétique ou par récupération des matériaux [Touzanne, 2002]. Les destinations des composants recyclés donnent les *options de fin de vie du produit.*

### 1.2.1 La valorisation des composants

Le recyclage par récupération des sous-ensembles correspond à une réutilisation ou à une récupération de certains constituants encore fonctionnels. La réutilisation est l'option de fin de vie la plus valorisante. Souvent les composants ont des durées de vie plus élevées que celle du produit entier. Par réparation de ces composants on récupère leur potentiel matériel et fonctionnel. Par exemple les circuits électroniques encore fonctionnels récupérés par le désassemblage des ordinateurs peuvent être utilisés dans la construction des jouets électroniques. Dans le cas où les sous-ensembles ou le constituant ne seraient pas fonctionnels, le recyclage par revalorisation des matériaux et le recyclage par revalorisation énergétique doivent être pris en considération.

### 1.2.2 La valorisation des matériaux

Le recyclage des matériaux a pour but la récupération des matériaux qui peuvent être utilisés en fabrication comme matières secondaires (matière première recyclée). Pour la revalorisation des matériaux les composants doivent être séparés par des opérations comme la refonte ou le broyage de la matière. La principale difficulté dans ce cas provient de la grande diversité de matériaux de ces composants et de la présence de polluants dans la matière. Dans ce contexte il faut analyser d'abord les profits obtenus par revalorisation des matériaux et les coûts des différents procédés de traitement.

### 1.2.3 La valorisation énergétique

Si la valeur des matières premières et des matériaux est inférieure au coût des opérations de traitement, le recyclage ne se justifie pas. La seule solution envisageable reste la valorisation énergétique ou la mise en décharge. Par la valorisation énergétique la matière se transforme en énergie par incinération. L'incinération a besoin d'une fraction de matières dont la valeur calorique dépasse une certaine valeur nécessaire pour récupérer l'énergie. Sinon, aucune récupération d'énergie n'est possible. Ce processus se confronte aussi au problème de la présence de substances non combustibles. La mise en décharge est la dernière option de fin de vie pour un produit et elle intervient quand la récupération fonctionnelle, matérielle ou énergétique n'est plus possible.

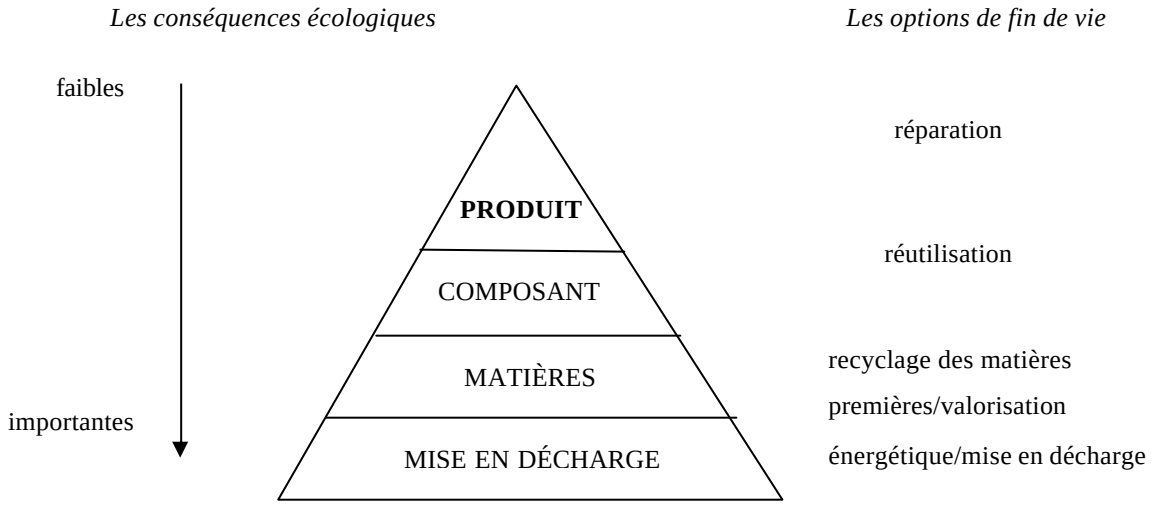



Fig. 3. Pyramide des options de fin de vie (après Hiessl et all. 1995)

Un produit en fin de vie peut être mis en décharge à condition qu'il ne contienne pas de polluants ou de substances nocives pour l'environnement. La figure 3, présente la pyramide des options de vie qui montre les niveaux d'application des options de vie et leurs conséquences écologiques. Plus le niveau de recyclage est haut, plus on a une production écologique.

L'évaluation écologique des options de fin de vie doit tenir compte de l'ensemble des impacts environnementaux. L'évaluation doit prendre en considération les effets négatifs comme les émissions, les déchets, les polluants, ainsi que les effets positifs correspondants aux opérations de recyclage.

Le recyclage par réutilisation totale des composants est un concept théorique puisque le produit en fin de sa vie est un produit usagé et il faut tenir compte de son état. Il peut présenter des modifications ou des altérations physiques qui le rendent impropre au démontage total des pièces et à leur réparation.

De même, la valorisation uniquement des matières ou énergétique n'est pas satisfaisante du point de vue économique. Il faut donc faire un compromis entre les trois types de

recyclage pour obtenir une valorisation optimale. Cette valorisation optimale est réalisée pendant un processus de recyclage bien mis au point. Au cœur du ce processus se trouve l'étape de désassemblage.

Le désassemblage est l'étape principale dans le processus de recyclage d'un produit en fin de vie. Le désassemblage, comme processus, a pour objectifs principaux l'extraction des sous-ensembles réutilisables, la séparation des matériaux, l'extraction des matières dangereuses.

Les produits à désassembler sont des produits usagés donc il faut tenir compte de leur état lors du désassemblage. Après leur collecte les produits sont soumis à un tri et sont diagnostiqués pour établir les options de fin de vie. Pour chaque produit, le scénario de valorisation est déterminé [Touzanne, 2002] [ Gerner, 2002]. Les produits sont désassemblés et/ou démantelés en fonction de ce scénario.

Le désassemblage permet d'une part de récupérer les sous-ensembles et les constituants réutilisables et d'autre part le tri des matériaux et l'extraction des matières dangereuses. La matière valorisée devient matière première recyclée et avec les pièces réutilisables donne les matériaux secondaires pour une nouvelle production.

Le flux des activités de recyclage est présenté à la figure 4.

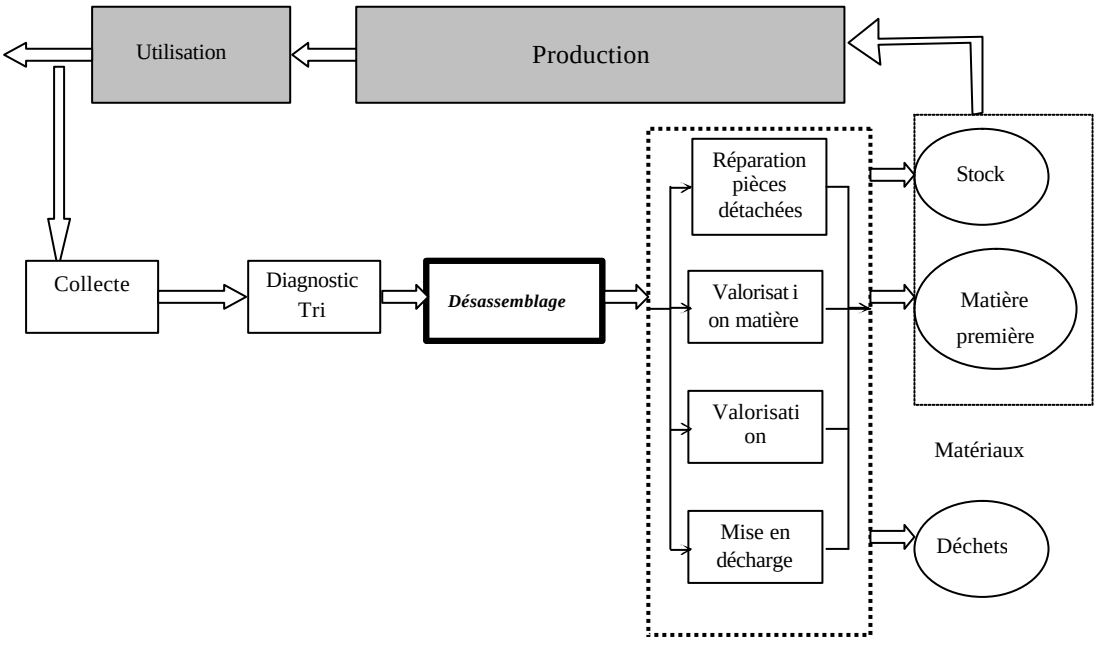


Fig. 4. Le flux des activités de recyclage pour un produit en fin de vie.

### 1.3 Clasifications du processus de désassemblage

La première étape dans la valorisation et la transformation d'un produit en fin de vie est donc le démontage de ce produit.

Le désassemblage comme processus de revalorisation doit permettre l'extraction des sous-ensembles et constituants réutilisables, la séparation des matériaux et l'extraction des matières dangereuses. Le scénario de désassemblage est lié à l'état du produit et aux options de fin de vie des composants de même que les opérations de désassemblage de ce scénario [Gerner, 2001].
Le désassemblage est une technique non-destructive. Le désassemblage, comme processus industriel, a pour objectifs principaux l'extraction des sous-ensembles réutilisables, la séparation des matériaux, l'extraction des matières dangereuses.

Si les composants ne sont pas réutilisables on applique des opérations destructives (partielles ou totales) : perçage, découpage, extraction, broyage ou différentes techniques thermiques. Ces techniques sont utilisées en vue de la récupération énergétique ou matérielle.

Dans ce contexte, trois types d'approches relatives au désassemblage peuvent être prises en compte : démontage total, partiel et démantèlement.

L'approche **démantèlement** repose en général sur le désassemblage destructif et il est envisagé dans le cadre du recyclage d'un produit usagé. Le but est donc de recycler certains composants, certaines matières ou de neutraliser tout simplement le produit. Il faut aussi noter que le désassemblage destructif peut être envisagé dans le cadre du désassemblage complet ou du démontage partiel lorsqu'il est possible de provoquer des dégradations du produit pendant ces processus. Ainsi il peut être plus rentable de dégrader ou de détruire certains composants du produit qui sont peu coûteux, plutôt que de mettre en oeuvre une procédure de démontage non destructrice et qui risque d'être longue et difficile à réaliser.

Les techniques de désassemblage actuelles font appel aux procédés mécaniques, chimiques ou thermiques. Parmi les techniques de désassemblage mécanique il y a l'extraction et le dévissage. Le perçage, le découpage, l'arrachage, la déformation mécanique ou thermique, les bains chimiques sont des procédés plus ou moins destructifs.

Certains auteurs prennent en considération **le désassemblage partiellement** ou **totalement destructif** [Gupta, 1996].

La dégradation et la déformation de certains composants, l'absence d'un constituant, le changement d'une solidarisation ou la présence de la corrosion sont des perturbations souvent rencontrées dans le processus de démontage. Il s'agit donc d'établir si on utilise des opérations destructives ou non dans le processus de démontage. Bien sûr, le désassemblage totalement destructif ne permet la récupération de constituants qu'au niveau de la matière ou de l'énergie.

Une classification plus approfondie du désassemblage a été proposée par F. Touzanne dans sa thèse [Touzanne, 2002]. Dans une première étape, l'auteur fait une classification par rapport au but du processus. Ainsi, il distingue deux types de désassemblage partiels : **le désassemblage sélectif** et **le désassemblage ciblé.**

Le but du désassemblage sélectif est d'obtenir des sous-ensembles fonctionnels sans chercher à obtenir les constituants élémentaires du produit (les pièces).

Le désassemblage ciblé a pour objectif l'extraction dans de bonnes conditions d'un ou de plusieurs constituants du produit. Dans ce cas, il est possible que le produit ne soit pas entièrement décomposé.
Par ailleurs, l'auteur fait une classification du processus de désassemblage d'après la technique utilisée et l'organisation du système. On a ainsi **le désassemblage linéaire** et **le désassemblage parallèle.** Le désassemblage est dit linéaire si un seul constituant final est désassemblé à la fois. Le désassemblage est parallèle si plusieurs constituants sont désassemblés en même temps.

Un autre mode de classification des processus de désassemblage est fait par rapport à la nécessité de démontage d'un composant avant un autre composant ciblé. De ce point de vue on a **le désassemblage monotone**, quand le désassemblage d'un constituant nécessite le désassemblage complet des autres constituants et **non monotone** quand on doit modifier la structure du produit pour accéder à un constituant puis le remettre dans l'état initial (par exemple enlever puis remettre un couvercle pour accéder à des constituants).

Jusqu'à présent, **le désassemblage** comme processus industriel est plutôt **manuel** ou **partiellement automatisé** que complètement **automatique**. Cette situation est due au fait que le système de désassemblage automatique est encore cher à réaliser par rapport aux revenus obtenus après la récupération des composants désassemblés, notamment du fait de la très grande flexibilité requise par le processus de désassemblage.

Ainsi, le désassemblage peut être classifié d'après plusieurs critères (cf. Fig. 5).

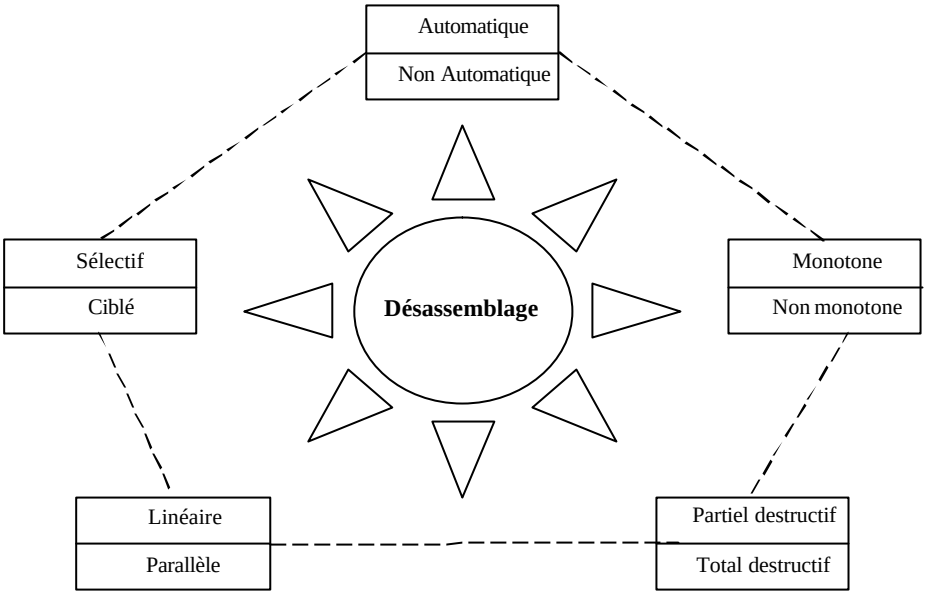



Fig. 5. Classifications du désassemblage

### 1. 4. Les spécificités du processus de désassemblage

Il y a des différences essentielles entre les problématiques de l'assemblage et du désassemblage. La première est que le désassemblage est généralement partiel ce qui ajoute une variable de décision très importante : la profondeur de désassemblage. Par ailleurs, en désassemblage il existe une incertitude très forte sur l'état des composants, c'est-à-dire sur leur valeur et sur la difficulté opératoire pour les récupérer. Enfin, il y a une forte variation sur la demande pour les composants récupérables, donc sur leur valeur.

Pour déterminer la meilleure stratégie de désassemblage il faut trouver la gamme optimale et sa profondeur. En effet, alors qu'en assemblage il faut assembler totalement les produits à réaliser (si on considère le conditionnement final comme assimilable à de l'assemblage quand certains composants sont emballés mais non assemblés au produit), en désassemblage selon le revenu estimé des composants ou de la matière récupérée et le coût des opérations de récupération, il n'est généralement pas rentable de procéder à un désassemblage total. Le choix d'un processus de désassemblage revient en fait, comme pour l'assemblage, à celui d'une gamme, d'une affectation des tâches à réaliser aux postes et d'un séquencement des produits à traiter. Toutefois dans le cas du désassemblage, la profondeur de la gamme est à choisir en fonction de la valeur des constituants et de la matière récupérable, valeur qui varie en fonction du marché, de l'état des produits traités (état qui se découvre pour partie lors du processus de désassemblage), et de la charge des postes à chaque instant.

Une *gamme de désassemblage* est une suite d'opérations effectuées en série et/ou en parallèle qui débute à partir du produit à désassembler et qui s'achève quand tous les constituants désirés sont obtenus. Un système de désassemblage est représentable par

une boîte noire (cf figure 6) ayant en entrée *p* flux $F_i^e$ ( i=1,..,p), chacun portant sur un ensemble d'objets matériels $\mathbf{P_i}$ tous identiques, et en sortie *q* flux $F_j^s$ ( j=1,..,q), chacun portant sur un ensemble d'objets matériels $\mathbf{C_j}$ également tous identiques [Touzanne, 2002].

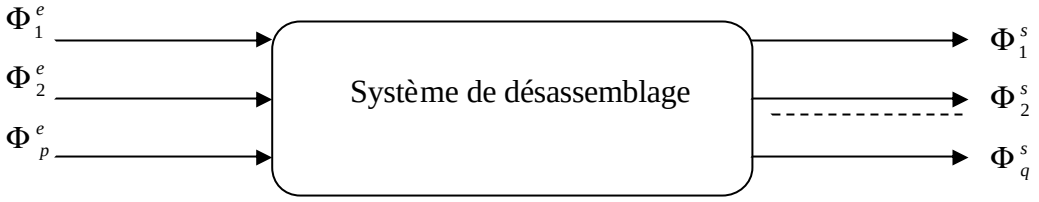



Fig. 6. Système de désassemblage

L'ensemble des opérations qui engendrent un flux $\Phi_j^e$ est appelé **processus de désassemblage**.

Les objets $\mathbf{P_i}$ sont appelés produits d'entrée, ils peuvent être des produits récupérés ou des sous-ensembles provenant d'un système de désassemblage amont. Les objets $\mathbf{C_j}$ sont appelés constituants élémentaires. Un constituant élémentaire est soit un composant élémentaire (une "pièce") soit un sous-ensemble destiné à la revente ou à un désassemblage plus complet au sein d'un autre système de désassemblage. Pour chaque composant il y a plusieurs possibilités de démontage. Une gamme de désassemblage définit l'ordre des opérations de démontage.

De même qu'en assemblage, la recherche de l'ensemble de désassemblage conduit à une explosion combinatoire des solutions à la suite du grand nombre de possibilités de décomposer un produit. Pour éviter l'explosion combinatoire il faut introduire plusieurs critères et contraintes concernant le désassemblage dans le but de réduire la recherche dans l'espace des solutions.

Les cas les plus difficiles à analyser sont bien sûr les grands produits industriels comme par exemple les produits électroniques ou les voitures. Dans ce dernier cas, le nombre d'opérations de désassemblage est immense, en fonction de l'ensemble des options de fin de vie pour les sous-ensembles de la voiture. Si pour l'assemblage d'un tableau de bord il faut en principe 7 opérations principales, en analysant les possibilités de désassemblage de ce tableau a la fin de sa vie on obtient au moins 30 opérations possibles en fonction des options de fin de vie pour chaque sous-ensemble et dans le cas favorable quand tous les éléments sont démontables [Addouche, 2002].

Dans l'assemblage tous les composants sont neufs et standardisés alors qu'en désassemblage certains composants peuvent subir des modifications géométriques et fonctionnelles. Alors, l'échec du désassemblage doit être envisagé. Dans ce cas on recherche des solutions alternatives de démontage totalement ou partiellement destructif, en tenant compte de toutes les dégradations qui peuvent affecter les liens entre les composants du produit. Parfois, les opérations destructives sont préférées aux

opérations de désassemblage. C'est le cas des produits qui ont parmi leurs composants des circuits imprimées, ou des matériaux récupérables par des traitements thermiques.

Par rapport au processus d'assemblage, le désassemblage est soumis à des contraintes économiques et techniques particulières. Pour un produit en fin de vie les caractéristiques géométriques, les fixations des pièces, leurs fonctionnalités ne sont plus les mêmes qu'au début de sa vie. Les contacts, les solidarisations, les relations entre les composants ont changé à la suite de leur utilisation.

Il faut avoir beaucoup d'informations sur l'état des liaisons entre les composants, sur la rigidité de ces liaisons, parce que l'extraction de certains composants peut être bloquée par des composants non directement liés mais très détériorés à la suite de l'utilisation du produit.

La dimension temporelle est beaucoup plus complexe à maîtriser dans le cas du désassemblage. Le temps nécessaire pour exécuter une opération de démontage est imprécis. Ce temps est lié à l'état du composant, au type de la fixation et à sa rigidité.

Le temps nécessaire pour le désassemblage de certains composants est différent de celui pour leur démantèlement. C'est-à-dire que le temps d'une opération de désassemblage est dépendant du type de désassemblage : destructif (partiel ou total) ou non destructif. A contraire, dans le processus d'assemblage les temps opératoires sont déterminés et constants, aux fluctuations près dans le cas de l'assemblage manuel, fluctuations qui restent toujours assez faibles.

A la technique de démontage employée et aux temps opératoires sont liés les coûts du désassemblage. Le critère économique évalue la profitabilité d'un processus de désassemblage. Une solution optimale par rapport à ce critère maximise la différence entre les revenus du recyclage et les coûts des opérations de désassemblage.

Mais si une solution profitable ne respecte pas les contraintes des lois environnementales, une telle solution n'est pas optimale par rapporte au critère écologique. Il est donc nécessaire d'évaluer les activités dans un processus de désassemblage en fonction de deux critères : économique et écologique [Gerner, 2001]. L'évaluation garantit une allocation efficace des ressources de recyclage à partir d'une justification économique et écologique. Par conséquent, une activité de désassemblage n'est appliquée que si elle est plus efficiente que des activités associées à d'autres techniques. Le critère économique analyse les revenus et les coûts. Une solution optimale par rapport à ce critère maximise la différence entre les revenus de revente des pièces récoupérées et des matières réutilisables d'une part, et les coûts de désassemblage, de broyage et de mise en décharge d'autre part. Le critère écologique estime les émissions dans le processus de consommation de l'énergie et de la matière. Une solution optimale par rapport à ce critère détermine un niveau de désassemblage dont les émissions sont minimales. Une solutions optimale par rapport au critère économique n'est pas forcément optimale par rapport au critère écologique et inverse. S. Gerner a exemplifié dans sa thèse comment les deux critères peuvent être accomplis dans le même processus de désassemblage [Gerner, 2001].

Le but du désassemblage est de minimiser ces coûts et de maximiser les revenus obtenus par la revalorisation des matériaux ou des pièces. Les revenus dépendent de l'état du composant après son démontage, de ses fonctionnalités et de son prix sur le marché secondaire. Les revenus et les temps opératoires sont en relation inverse: plus longue est la séquence de démontage, plus la possibilité d'obtenir des composants en bon état diminue. Il faut donc connaître *la profondeur de désassemblage* pour maximiser le bénéfice de ce processus [Chevron, 1999]

Du point de vue économique un processus de désassemblage mal conçu entraîne des pertes.

Si on considère un produit qui est désassemblé en sous-ensembles et en composants, la courbe qui représente le revenu obtenu par la récupération ou par le recyclage du sous-ensemble ou composant augmente linéairement [Johnson, 1998] (figure 7.) Il y a deux extrêmes: la situation B dans laquelle le composant ou le matériel récupéré a une valeur basse et la situation A quand le produit est en bon état et on récupère le maximum de la valeur de composant à chaque étape de désassemblage.

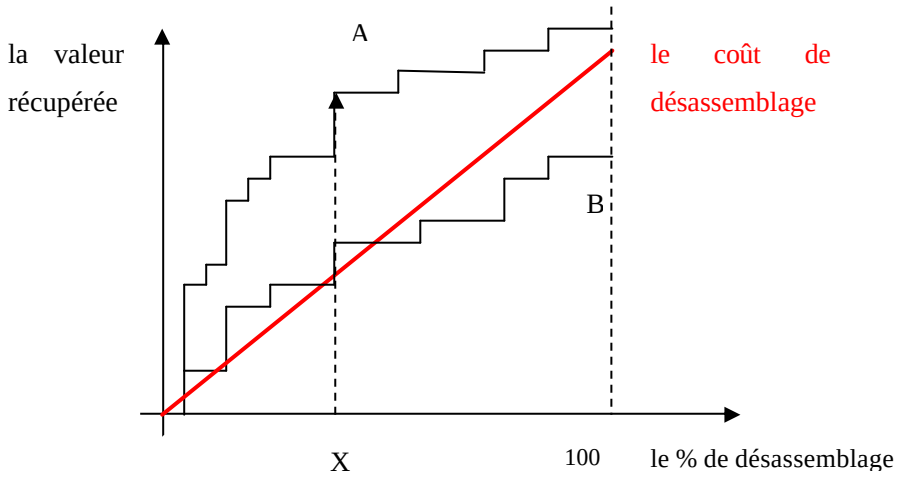



Fig. 7. Maximiser le profit dans le processus de désassemblage

Pour maximiser le profit du processus de désassemblage il faut tenir compte de la conception pour le désassemblage et de l'optimisation des séquences. Dans la figure 7 le point X est défini comme le *point cible* où le bénéfice économique est maximal.
Dans un système d'assemblage les composants convergent vers une seule source : le produit final. Le principe qui conduit ce système est la convergence. Au contraire, dans un système de désassemblage chaque composant, part du produit initial, en fonction de demande. De plus, les opérations de désassemblage dépendent de la demande par les différentes constituants, donc de leur valeur. L'assemblage est un processus convergent pendant que le désassemblage est *un processus divergent* [Kizilkaya, 1998].

### 1. 5. Comparaison entre les problématiques de l'assemblage et du désassemblage

La conception et la commande des systèmes de désassemblage des produits en fin de vie ne peut pas être simplement considérée comme symétrique de celle de la conception des processus d'assemblage, à laquelle de nombreux travaux, dont ceux du LAB ont été consacrés depuis une vingtaine d'années. Il existe des différences importantes et bien connues, principalement: l'incertitude sur l'état des produits traités, leur très grande diversité, la nécessité de recourir à des opérations alternatives et partiellement destructives en cas d'échec des opérations de récupération des constituants et la variabilité des temps opératoires.

### 1.5.1. Conception

La structure d'un système de production est construite à partir d'un modèle de processus. Ce modèle représente soit l'ensemble des gammes de fabrication, soit l'affectation des tâches sur les machines et l'ordonnancement des opérations.

Concevoir un système de production implique la réalisation de grandes étapes comme: la modélisation du produit, la génération du processus, le représentation des processus, l'optimisation, le pilotage du système, la planification de la production et le découpage en postes.

Ces étapes conduisent à l'obtention de résultats intermédaires tels que: les gammes de désassemblage, l'organisation des équipements, l'affectation de ressources ou la commande du système.

#### 1.5.1.1. Conception des systèmes d'assemblage

Physiquement, un système d'assemblage est constitué d'un ensemble de ressources (opérateurs humains, robots, convoyeurs) agencées spatialement, capables d'assembler selon une cadence imposée un produit ou plus généralement une famille de produits. La conception d'un tel système suppose définies l'ensemble des tâches à effectuer ainsi que les données de production pour pouvoir proposer des équipements capables d'effectuer ces tâches au niveau de performance requis par les prévisions de production et définir leur organisation spatiale.

La démarche de conception consiste donc d'abord à définir les tâches à accomplir, ce qui nécessite de partir du produit (ou de la famille de produits) à assembler, et de déterminer une séquence d'opérations (gammes) permettant de construire le produit final à partir de ses composants élémentaires. Une structure admissible pour le système d'assemblage recherché consiste à partir d'une gamme admissible et à la découper en un certain nombre de groupes d'opérations qui constitueront les postes du système, placés en série et/ou en parallèle.

En fait, il s'avère que pour un produit donné l'ensemble des gammes admissibles est généralement très élevé et que pour chaque gamme, le nombre de découpages en postes capables de satisfaire la production prévues est également très élevé. Le problème de la conception devient ainsi une recherche de la solution optimale dans le problème à deux

étapes que nous venons d'esquisser brièvement. Une solution est optimale si elle minimise le coût unitaire d'assemblage qui est en première approximation composé d'un coût fixe divisé par le nombre de produits et d'un coût variable par produit. Ce coût n'est pas calculable à partir de la connaissance des gammes seules, il faut aller jusqu'à l'étape de définition des postes, ce qui fait que contrairement à ce qui a pu être écrit parfois, dans la phase de conception d'un système d'assemblage le concept de gamme optimale n'a généralement pas de sens. Par ailleurs, dans le cas d'un système fortement manuel il est également important d'équilibrer la charge des postes. Dans les systèmes fortement automatisés, le système optimal conduit généralement à un bon équilibrage, mais pas nécessairement à l'équilibrage optimal.

La conception d'un système d'assemblage suppose tout d'abord la définition du niveau d'automatisation. Ce choix, basé sur la quantité de produits à fabriquer et le retour sur investissement a largement été étudié, notamment par Boothroyd et Dewhrust [Boothroyd, 1986], nous le supposerons établi. Dans le cas considéré ici, l'automatisation est nécessairement flexible donc basée sur l'emploi de robots. Nous distinguerons deux situations :

- Les systèmes robotisés pour lesquels l'objectif de la conception, pour un niveau de productivité, de qualité et de fiabilité donnés revient essentiellement à minimiser les coûts d'investissement devant lesquels les coûts de fonctionnement peuvent être négligés en première approximation.
- Les systèmes manuels pour lesquels la situation est inversée et où ce sont les coûts de fonctionnement qu'il convient de minimiser, ce qui revient à équilibrer le système, c'est-à-dire équilibrer la charge des postes, problématique bien connue sous le nom d'Assembly Line Balancing (ALB).

**Notons :**

$T$ : le temps d'assemblage d'un produit, c'est-à-dire, en première approximation la somme des durées opératoires des opérations permetant de le réaliser
$T_c$ : temps de cycle requis (temps moyen séparant la sortie de deux produits du système)

La contrainte de productivité s'exprime par :

$$(n-1)\cdot T_c \leq T \leq n \cdot T_c \qquad (1)$$

où n est un entier $\geq 1$.
On définit ainsi un ensemble de $n$ postes d'assemblage qui constituent une solution valide.

Il reste à rechercher parmi l'ensemble des solutions valides une solution optimale ou, plus raisonnablement un ensemble restreint à quelques solutions quasi optimales pour permettre une marge de décision à l'équipe de conception du système. Les méthodes permettant de rechercher ces solutions sont bien connues, notamment les algorithmes stochastiques qui permettent de traiter des problèmes de grande taille.

L'organisation la plus générale résultante est une ligne de *n* postes, en série et/ou en parallèle, les deux organisations aux extrémités des possibilités étant la ligne complètement série et la mise en parallèle de *n* postes identiques.

La mise en parallèle est avantageuse pour ce qui concerne la fiabilité, puisque l'arrêt d'un poste ne perturbe pas le fonctionnement des postes placés en parallèle, alors que des postes en série dépendent du bon fonctionnement des postes amont et aval. On peut découpler les postes série par des buffers, mais au prix d'une surface requise plus grande et d'un encours moyen augmenté et, de toute façon, la capacité de ces buffers ne permet de faire face qu'à des arrêts de courte durée.

La structure parallèle serait donc idéale, mais elle est rarement utilisée pour les raisons suivantes :

1. elle nécessite la duplication des outillages,
2. elle nécessite la duplication des alimentations en composants, cette duplication entraîne un surcoût, faible en manuel, très élevé en automatique, de plus le nombre d'alimentations dans un poste est limité pour des raisons évidentes de place.
3. en manuel le temps opératoire moyen croît avec le nombre d'opérations différentes à effectuer du fait d'une moins grande répétitivité des tâches.

Pour ces raisons on privilégie généralement la structure série, réservant la parallélisation uniquement pour des opérations qui dépassent le temps de cycle.

Cela étant posé, dans le cas d'une organisation série (en ligne), la définition structurelle du système se ramène essentiellement à définir une suite d'opérations à effectuer, totalement ou partiellement ordonnées, et à découper cette suite en *n* groupes de tâches $G_i$ ($i = \overline{1..n}$), sous la contrainte qu'aucune opération du groupe $G_i$ ne précède une opération d'un groupe $G_j$ tel que i>j. A ce stade on peut distinguer deux méthodes selon qu'on travaille sur des suites totalement ordonnées (gammes d'assemblage) ou des suites partiellement ordonnées représentables, par exemple, par des graphes de précédence. Dans le premier cas on recherchera pour chaque gamme la configuration optimale puis la meilleure des configurations obtenues et dans le second on aura directement la configuration optimale. Dans les deux cas il convient donc de déterminer préalablement l'ensemble des gammes admissibles et, pour la seconde approche la représentation collective correspondante.
Nous nous placerons dans le cas de la seconde approche et supposerons résolu le problème de l'obtention de l'ensemble des gammes admissibles et de sa représentation collective, en l'occurrence un graphe de précédence. Ces problèmes ont été traités dans les travaux antérieurs du LAB. Une structure potentielle du système d'assemblage à concevoir est alors donnée par une partition du graphe de précédence en n sous-graphes respectant la contrainte de précédence rappelée plus haut.

Lorsque la production multiproduits est organisées en lots, pour chaque lot les temps opératoires sont bien établis et on obtient une solution qui sera proche du

fonctionnement effectif. Dans le cas où la production est basée sur un mélange des produits, il faut considérer des temps opératoires moyens, au prorata des proportions de produits prévues et dans ce cas le comportement effectif du système obtenu fluctuera considérablement autour du fonctionnement moyen prévu.

1.5.1.2. Conception des systèmes de désassemblage des produits en fin de vie

La conception d'un système de désassemblage est un processus complexe qui nécessite la détermination de nombreux résultats. Il existe peu de démarches globales regroupant l'ensemble des aspects de conception. Dans la littérature plusieurs méthodes pour la conception ont été proposées: l'évaluation et l'optimisation du processus de désassemblage, la planification et la définition des stratégies de désassemblage, la détermination de la gamme optimale, la détermination des options de valorisation appropriées, l'estimation des coûts et des temps associés à la gamme. [Touzanne, 2002].

La problématique de conception des systèmes de désassemblage est fondée sur les mêmes considérations que la problématique de conception des systèmes d'assemblage, elle en diffère néanmoins sur les point suivants :

1. le désassemblage n’est pas total, certains sous-ensembles seront simplement broyés pour permettre le tri et la récupération des matières qui les composent,
2. la variabilité des types de produits traités et donc la flexibilité requise est beaucoup plus élevée.
3. l’état des produits est généralement très incertain, ce qui introduit une grande variabilité des temps opératoires et aussi la nécessité de prévoir des opérations alternatives destructives pour extraire certains composants ou sous-ensembles,
4. la demande sur les pièces récupérées est variable ce qui implique une variabilité dans les destinations fin de vie : réutilisation ou récupération matière, dans le premier cas il faut démonter soigneusement, dans le second on peut sans problème détériorer volontairement ces sous-ensembles pour gagner du temps.
5. la fonction à optimiser dans le démontage est composée du revenu de ce qui est récupéré (pièces ou matières) et du coût de démontage, il s'ensuit que le niveau de démontage est une décision du système de commande, variable suivant l'état du marché des matières et des pièces récupérées.

La conception des systèmes de désassemblage est globalement basée sur la même démarche que celle utilisée pour les systèmes d’assemblage, elle consiste essentiellement à partir des produits pour en déduire l’ensemble des gammes admissibles, séquences dont le partitionnement définit l’ensemble des systèmes admissibles. La problématique diffère toutefois sur les quelques points indiqués ci-dessus. Conceptuellement, la différence la plus importante tient dans la non imposition du niveau de désassemblage dans le second cas et de la possibilité de processus alternatifs, selon qu’on choisit un démontage propre pour récupérer des composants ou un démontage destructif.

Alors, a partir de la prévision de production, de données économiques: (revenus des constituants et de la matière récupérée, coût de la mise à la décharge, coûts opératoires),

de l'ensemble de gammes de désassemblage admissibles la conception des systèmes de désassemblage est un processus complexe à développer.

Ici on doit souligner que les coûts opératoires sont simples à évaluer si le système est monoposte (un poste désassemble tout le produit mais il faut avoir n postes en parallèle) Ces coûts sont proportionnels aux temps opératoires. Pour les estimer il faut déterminer l'ensemble des gammes admissibles, donc l'ensemble des opérations (on doit pouvoir s'en tenir à des opérations génériques comme dans les graphes de précédences). Il faudrait aussi évaluer une probabilité de réussite des opérations pour déterminer un coût moyen. Si le système est en ligne, le coût va dépendre de l'équilibrage ou mieux du débit de la ligne et donc des circonstances.

### 1.5.2. Conduite

La conduite peut être considérée comme l'art d'adapter en permanence les objectifs de l'entreprise à l'évolution de l'environnement à travers l'analyse des contraintes et des opportunités. La conduite d'un système de production se définit ainsi comme la capacité à répondre à une demande fluctuante en nature et en quantité de produits variés et personnalisés, dans des délais respectés et de plus en plus courts.

C'est une démarche visant à répondre à la demande du client et qui correspond à l'un des facteurs mesurant la compétitivité industrielle de l'entreprise. La conduite d'un système de production consiste à utiliser un système de décision pour faire exécuter par le système physique l'ensemble des opérations de fabrication qui lui sont affectées.

La conduite en temps réel d'un système de production est une tâche complexe qui requiert des connaissances dans les domaines de l'informatique, de l'automatique, de la production, de la communication homme – machine. La conduite dans ce cas signifie la surveillance et la commande des paramètres du système. Il s'agit d'une action corrective, prédictive ou réactive de certaines variables.

#### 1.5.2.1. Commande des systèmes d'assemblage

La commande des systèmes d'assemblage, comme toute commande de système complexe est hiérarchisée en plusieurs niveaux :

***Planification :*** La planification fournit l'ensemble des produits à assembler par type de produits et par période, généralement la semaine, sur un horizon T, généralement de quelques mois. Elle résulte du placement dans le temps des produits à assembler, à partir des commandes enregistrées et de prévisions. Ce placement tient compte de la capacité et suppose un lissage des charges. Il peut, notamment, faire l'objet d'une optimisation fournissant le meilleur compromis entre la constitution de stocks dans les périodes sous chargées et le recours à des heures supplémentaires dans les périodes surchargées. C'est à ce stade qu'est défini l'équilibrage de la ligne, donc l'organisation des postes.

***Ordonnancement***: L'ordonnancement définit, le séquencement (c'est-à-dire l'ordre de passage des produits sur la ligne). A ce stade, on cherche généralement à optimiser l'équilibrage de la ligne ou encore à optimiser l'équilibrage et à minimiser le retard entre la date de sortie des produits et la date à laquelle ils sont requis. Plusieurs travaux de recherche ont par ailleurs proposé des méthodes de réaffectation dynamique des tâches aux postes. A notre connaissance il n'y a pas eu d'application de ces méthodes, un problème concret majeur qui en limite la portée réside dans le fait que les alimentations en composants figent les affectations.

***Pilotage*** : Pour les systèmes automatisés tout ou partie, le pilotage envoie les ordres aux équipements.

Nous nous intéressons ici aux niveaux ordonnancement. Dans les deux cas la problématique rejoint en partie celle de la conception, en effet, il faut définir l'ordre d'éxécution des opérations ainsi que leur affectation à des postes, on retrouve bien les deux temps : choix d'une gamme et découpage de cette gamme en postes. La différence, conceptuellement faible mais pratiquement très importante est que dans le cas de la commande, le système est défini et les choix possibles sont très réduits.

#### 1.5.2.2. Commande des systèmes de désassemblage

##### *Niveau Planification*

La planification du travail dans un système de désassemblage relève en partie de la même problématique que celle de tout système de production manufacturier, à partir d'un plan de travail défini sur plusieurs mois. On établit un plan de charge semaine par semaine en réalisant un compromis entre le niveau de stockage résiduel des produits non démontés en fin de chaque période pour laquelle la charge dépasse la capacité nominale et l'accroissement de la capacité par le recours à des heures supplémentaires.

Toutefois, là encore, le niveau de démontage est un moyen d'action supplémentaire. En effet, il a été défini dans la phase de conception de la ligne, mais sur la base d'un coût horaire normal. Dans le cas d'emploi d'heures supplémentaires, le niveau de démontage optimum précédemment calculé peut être remis en question. Il s'agît ici d'optimiser le gain sur la période considérée, en jouant sur deux facteurs de coûts : le coût horaire, comportant éventuellement des heures supplémentaires, et le coût de stockage des produits non traités et le revenu des éléments récupérés (coût éventuellement négatif pour les éléments non récupérés et partant en décharge). Nous avons choisi dans ce travail de ne pas traiter ce problème et de ne considérer que les niveaux ordonnancement.

##### *Niveau Ordonnancement*

L'objectif de l'ordonnancement est de traiter l'ensemble des produits programmés sur la période considérée en optimisant le revenu effectif résultant du revenu apporté par les éléments récupérés diminué du coût de démontage.

La très grande variabilité des temps opératoires, et plus encore la variabilité opératoire (démontage propre ou destructif) qui se découvre pour une bonne part en temps réel rend la qualité de cet ordonnancement extrêmement incertaine et nécessite une grande réactivité du système, ce qui donne une importance particulière à la mise en œuvre d'un ordonnancement dynamique. De plus, outre l'affectation des tâches aux postes et le séquencement, il existe deux éléments de décision importants qui n'existent pas dans le cas de l'assemblage :

- le niveau de désassemblage,
- la méthode de démontage : propre pour récupérer les composants ou destructive pour récupérer la matière.

Ces deux éléments sont corrélés, on désassemble d'autant plus qu'on cherche à récupérer davantage de pièces en bon état ce qui requiert un démontage soigné.

### 1.6. Désassemblage – Analyse Bibliographique

Ce paraghraphe est une analyse bibliographique sommaire où on présente une classification de tous les travaux (thèses ou articles) qui traitent le domaine du désassemblage en vue de recyclage. Les travaux sont distribuées dans 4 catégories en fonction du niveau auquel on fait l'étude du désassemblage : niveau conception, niveau planification, niveau d'ordonnancement désassemblage et le niveau commande.

#### 1.6.1. Conception des systèmes de désassemblage

La plupart des recherches ont eu pour objectif la détermination des gammes de désassemblage. Les recherches développent de nouvelles méthodes pour l'amélioration et l'optimisation du processus de désassemblage, pour la conception d'outils et de robots pour le désassemblage. Dans [Touzanne, 2002], l'auteur a fait un bon état de l'art sur les travaux principaux relatifs aux différentes étapes nécessaires à la conception des systèmes de désassemblage.

Touzanne a classifié les travaux en trois catégories. La première concerne les différentes méthodes de conception des systèmes de désassemblage. Ces méthodes sont présentées dans le contexte de la valorisation du produit ou de ses composants, ou dans le contexte de la maintenance. Concevoir un système implique la génération des gammes de désassemblage, l'organisation et l'affectation des ressources, éventuellement la commande du système. Les auteurs ont proposé des méthodes différentes pour l'évaluation ou optimisation d'un processus de désassemblage. Johnson et Wang, Kopacek, Feldman, Zhang et Vujosevic ont traité ce sujet depuis les années '95-'96 [Touzanne, 2002]. Les objectifs de ces travaux ont été principalement : la définition de la stratégie de désassemblage, la génération et la représentation du processus, l'optimisation de la valorisation, la sélection des outils nécessaires à la réalisation du désassemblage, les lignes directrices pour la conception du produit.

La phase suivante est *l'analyse du produit*. Cette phase doit déterminer les composants et les matières à récupérer et si le produit peut être valorisé, établir la technique de désassemblage et déterminer les séquences de désassemblage. Ici le terme de "désassemblabilité" est pris en considération. Des auteurs comme Johasson, Bjorkman, Woo, Dutta, Srinivasan ont défini différèment cette notion [Touzanne, 2002]. La direction de désassemblage, les degrés de liberté et la trajectoire du composant désassemblé ont été étudiés.

La troisième catégorie est la plus étudiée. Il s'agit de la détermination des gammes de désassemblage. La génération des gammes est une des principales étapes de la conception d'un système de désassemblage. De nombreux auteurs ont traité ce problème. Leurs recherches diffèrent par les méthodes de modélisation des gammes. Lambert dans son étude [Lambert, 2003] a énuméré plus de 50 articles qui ont décrit des recherches sur le séquencement des opérations[1]. Des chercheurs connus comme Gungor, Gupta, Dini, Dutta, Zussman, Lambert, Delchambre, Moore, Lapérierre ont développé différents outils graphiques pour la description des gammes de désassemblage. Les arbres de désassemblage, les graphes ET/OU, les graphes de désassemblage, les Réseaux de Pètri de désassemblage, les graphes de contacts, les graphes de précédence, les graphes d'état constituent des outils graphiques souvent utilisés pour la représentation des gammes de désassemblage. Il n'existe pas une unité d'opinion pour ce qui concerne l'outil plus approprié pour cette opération. A l'heure actuelle un résumé des méthodes pour la détermination et la représentation des gammes de désassemblage est nécessaire ainsi qu'une classification des différentes approches proposées.

### 1.6.2. Planification du désassemblage

Le domaine de la planification du désassemblage inclut la recherche des gammes de désassemblage. Le but de la planification du désassemblage est de trouver la gamme optimale de désassemblage par rapport aux critères économiques. Les variables importantes dans la planification du désassemblage sont les coûts du processus et les revenus obtenus par la valorisation des composants désassemblés. Pour établir un plan optimal de désassemblage il faut un compromis entre les deux variables. Il y a des cas où la séparation des composants n'est pas possible ou des opérations de désassemblage qui ne sont pas profitables. Le problème de la planification du désassemblage devient ainsi très complexe.

En 1998 O'Shea, Grewal et Kaebernick ont élaboré un état de l'art de la planification du désassemblage [O'Shea, 1998]. Ils ont cité plus de 60 articles et travaux qui ont traité le problème de la planification du désassemblage. La plupart des auteurs ont considéré le désassemblage comme l'inverse de l'assemblage et ont élaboré des plans de désassemblage en se basant sur cette hypothèse. Par la suite, les chercheurs ont étudié la stratégie de désassemblage à partir de l'ensemble des gammes. Ils ont cherché la stratégie optimale entre tout jeter et tout récupérer après le désassemblage. L'accent a alors été mis sur la détermination de l'aptitude du produit à être valorisé. Alors, les

[1] Dans l'étude de Lambert il s'agit du séquencement des opérations de désassemblage des produits et pas du séquencement des produits sur la ligne de désassemblage.

recherches ont été orientées vers la capacité de récupération des composants et/ou des matières en vue du recyclage du produit. Des chercheurs comme Zussman, Laperrière, Gupta, Penev, Mascle, Moore ont tenu compte de la valeur de fin de vie du produit. Dans le chapitre suivant nous allons donner quelques exemples de travaux dans le domaine de la planification du désassemblage.

### 1.6.3. Ordonnancement du désassemblage

Dans le domaine de l'ordonnancement du désassemblage la littérature est sommaire. L'ordonnancement, en général, est un élément important dans l'ensemble des tâches liées au pilotage. Un ordonnancement consiste à organiser dans le temps la réalisation des tâches de désassemblage compte tenu de contraintes temporelles et physiques, de la disponibilité des ressources matérielles et humaines. Un ordonnancement décrit l'ordre d'exécution des tâches et l'allocation des ressources au cours du temps, afin de satisfaire un ou plusieurs critères d'optimisation.

L'ordonnancement du désassemblage consiste à affecter les de manière à équilibrer la ligne. Gupta, Taleb, Wiendahl, Kizilkaya sont des chercheurs qui ont travaillé sur le problème d'affectation des opérations de désassemblage. Par contre, l'équipe de Gungor et Gupta a travaillé sur l'équilibrage de la ligne de désassemblage (voir Chapitre 2). Néanmoins, leur approche est prédictive et non réactive. Il y a un manque d'étude en vue de l'adaptation des décisions prévues en fonction de l'état courant du système de désassemblage. C'est le motif pour lequel notre recherche est guidée vers l'étude de l'ordonnancement du désassemblage en fonction de plusieurs variables du système : la désassemblabilité et la valeur de fin de vie du produit, l'emplacement des ressources, le type du désassemblage (destructif ou non). La méthode proposée est décrite dans le Chapitre 4.

### 1.6.4. Commande des systèmes de désassemblage

Le problème de la commande du désassemblage est l'aspect le moins traité dans la littérature. La raison en est que peu des systèmes de désassemblage automatisés ont été mis en fonction jusqu'au présent. En effet, un système de désassemblage entièrement automatisé est coûteux, la valeur de fin de vie d'un type de produit ne pouvant pas couvrir le coût de la ligne. Dans ce sens, la flexibilité du système de désassemblage est une caractéristique importante et les chercheurs essayent de concevoir des lignes de désassemblage pour les familles de produits.

Les cellules de désassemblage semi-automatisées ou complètement automatisées sont formées soit d'un seul robot et d'une table rotative comme dans le cas de la cellule conçue par Puente [Puente, 2003], soit des plusieurs robots avec des tâches spécifiques et d'un opérateur comme dans la ligne décrite par Kopacek dans [Kopacek, 2003] ou par Gerner dans [Gerner, 2001]. Dans touts les cas le désassemblage est total et s'applique aux produits peu dégradés pendant leur vie (comme les ordinateurs ou les téléphones portables).

Au niveau de la commande il faut envisager des décisions en temps réel. En fonction de l'état du composant du produit on décide s'il est désassemblé ou pas. De même, le type de désassemblage est choisi : propre ou destructif. De tels types de décisions sont difficiles à prendre. L'incertitude dans le cas d'un produit usé est grande. L'état des composants n'est pas connu au début du désassemblage du produit. Si l'opération de désassemblage est bloquée soit du fait d'un composant détruit ou manquant, soit de déformations, la gamme de désassemblage doit être changée ou même il faut abandonner l'opération. Dans ces conditions le désassemblage sur une ligne automatisée est presque irréalisable.

Le problème de la commande a été étudié par des chercheurs qui ont proposé des modèles pour contrôler le flux sur la ligne de désassemblage. Chevron, Puente, Kopacek sont les trois chercheurs qui ont proposé des outils pour la commande et ont implémenté ces méthodes dans leurs cellules de désassemblage automatisées ou semi-automatisées. Les recherches à venir démontreront si un système automatisé de désassemblage est profitable ou non.

### 1.6.5. La distribution de travaux dans le domaine du désassemblage

Ce paragraphe présente une statistique des travaux sur le désassemblage des produits en vue de leur recyclage. On doit préciser que l'information est rapportée aux travaux, articles et thèses consultées par l'auteur de cette thèse jusqu'à présent et non au nombre total de travaux existants dans ce domaine. Le but est de présenter au lecteur la contribution des chercheurs aux diffèrents niveaux d'étude du désassemblage.

#### 1.6.5.1. Histogramme

Les premières études sur le désassemblage ont été publiées en 1989 dans l'article de A. Delchambre [Delchambre, 1989]. Par sa recherche, Delchambre et son équipe ont développé une base de connaissance en utilisant la planification du désassemblage. La première approche méthodologique pour la détermination des gammes opératoires a été présentée en 1990 par C. Mascle dans [Mascle, 1990]. Woo et Dutta ont appliqué la méthode de désassemblage en trois dimensions [Woo, 1991]. Lapperrière en 1992 a présenté un résumé de la planification d'assemblage par l'étude du désassemblage en vue de la maintenance [Lapperrière, 1992]. En 1994 un article complet sur l'ordonnancement du désassemblage est publié [Gupta, 1994]. Ses auteurs, Gupta et Taleb, ont proposé un algorithme d'ordonnancement du processus pour satisfaire la demande de composants.

Les années qui ont suivi, les publications ont principalement été centrées sur la détermination des gammes opératoires et la planification du processus de désassemblage lui-même. La période la plus riche en publications dans ce domaine a été entre les années 1997-1999. Le premier article sur l'équilibrage des lignes de désassemblage a été publié en 1999 [Gungor, 1999a]; ses auteurs Gungor et Gupta ont proposé une méthode heuristique pour minimiser le temps de cycle.

Le début du 21ème siècle est caractérisé par une diminution du nombre de travaux. Quand même, en 2002 la première thèse sur la conception des systèmes de désassemblage avec une bibliographie complète est due à F. Touzanne [Touzanne, 2002]. Dans les trois dernières années, le nombre de publications liées au domaine du désassemblage a augmenté. Les travaux sont orientés sur l'étude du désassemblage automatique. Les chercheurs présentent des réalisations des cellules automatiques de désassemblage. Torres et Puente en [Torres, 2004] presentent le désassemblage automatique d'un ordinateur fixe. Kopacek décrit le désassemblage semi-automatique des téléphones portables dans [Kopacek, 2005]. Par ailleurs, le spectre des méthodes pour la planification du processus de désassemblage est plus large : logistique inverse, réseaux de Petri flous, algorithmes heuristiques, algorithmes génétiques. Quand même, le domaine de l'équilibrage des lignes de désassemblage n'est pas encore suffisamment exploité. Notre approche se situe dans cette ligne de recherche. Dans la Figure 8. L'évolution du nombre des travaux sur le désassemblage publiés est représentée par années.

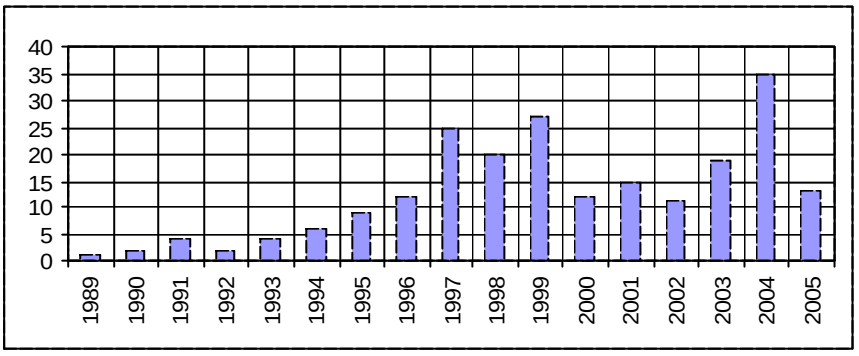



Fig. 8. Evolution du nombre des publications liées directement au désassemblage

1.6.5.2. Les niveaux de recherche

Comme on l'a déjà présenté, la plupart des travaux sont concentrés au niveau de la détermination des gammes de désassemblage et de la planification du processus de désassemblage. Beaucoup des méthodes ont été proposées pour la modélisation et la détermination des gammes : méthodes graphiques, numériques ou heuristiques. La partie la moins étudié d'un processus de désassemblage reste l'ordonnancement et la commande du processus. Dans la Figure 9 le pourcentage des travaux dans chaque champ d'étude est représenté.

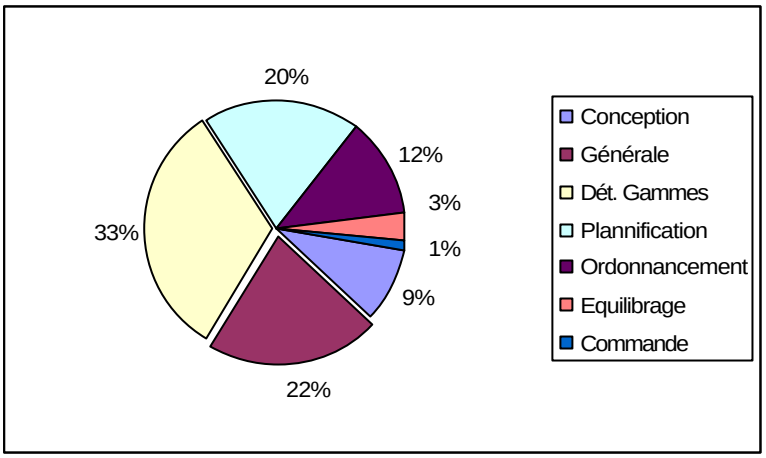



Fig 9. La repartition des travaux par rapport aux niveaux de recherche en désassemblage

Nous avons pris comme référence les travaux sur le désassemblage étudiés pendant cinq années. On estime que ces résultats ont un bon coefficient de précision, c'est à dire que la plupart des ouvrages existant dans le domaine du désassemblage des produits en fin de vie ont été consultés.

### 1.7. Conclusions

Ce chapitre est une présentation sommaire de la problématique du désassemblage. Dans les 15 dernières années, le nombre des recherches dans ce domaine a augmenté. L'intérêt porté au désassemblage est lié aux notions de recyclage et de récupération des composants et/ou de la matière des produits manufacturièrs en fin de leur vie. La nécessité de réutiliser la matière secondaire dans le processus de fabrication est évidente dans la société de consommation actuelle. Les nouvelles réglementations légales sur la protection de l'environnement donnent une plus grande importance aux étapes du processus de recyclage des matières récupérées par le désassemblage.

Dans le désassemblage, une des principales difficultés provient de la possibilité d'échec du démontage à cause du mauvais état du produit usé. Dans ce cas on doit choisir entre un désassemblage destructif et l'abandon du processus. Parfois, cette décision doit être prise en temps réel puisque l'état des composants n'est pas a priori connu. On peut dire que ce problème peut être modélisé comme un processus de prise de décision devant satisfaire un ou plusieurs critères d'optimum.

La plupart des recherches dans le domaine du désassemblage sont concentrés sur la planification du processus. De même, la détermination et la représentation des gammes de désassemblage constituent encore un sujet de grand intérêt pour les chercheurs du domaine. Toutefois, le nombre d'articles qui traitent l'ordonnancement du désassemblage est bas.

Il y a un manque des travaux en ce qui concerne le contrôle et la commande en temps réel des systèmes de désassemblage au niveau de l'ordonnancement et de l'équilibrage de la ligne. Ce manque est dû à l'implémentation difficile de la ligne automatisée de désassemblage. Souvent, de telles lignes ne sont pas profitables. Notre étude se situe dans les démarches en vue de l'optimisation du processus de désassemblage et propose une nouvelle approche pour la conduite des systèmes de désassemblage.

# CHAPITRE II

# PROBLÉMATIQUE DE LA COMMANDE

## 2.1. Structure hiérarchisée du système de commande

La commande d'un système de production est le processus qui assure la réalisation des opérations à la suite d'un plan de fabrication. A tous les niveaux du système de production la commande se manifeste par la surveillance et l'analyse de développement du processus de fabrication et par des actions correctives sur certains paramètres et variables. Ces actions sont assurées par le module de commande du processus. La commande est donnée à la suite d'une comparaison entre les valeurs prévues et le valeurs réelles de certains paramètres.

Compte tenu de la flexibilité des systèmes de fabrication, la commande intègre des aspects de surveillance et de décision. La surveillance doit effectuer différentes tâches, essentiellement la détection et le diagnostic des situations anormales. La décision donne une solution pour le traitement d'une situation indésirable.

La commande d'un système de production est hiérarchisée en plusieurs niveaux: **la planification**, **l'ordonnancement hors ligne**, **l'ordonnancement en ligne** et **le pilotage**.

### 2.1.1. La problématique de la planification

Une définition plus générale affirme que la planification est le processus par lequel on décide en avance *quoi* et *comment* faire. Le processus de planification entraîne la traduction des commandes et/ou prévisions de commandes dans des ordres de fabrication. Le plan peut être simple ou élaboré, il doit contenir étape par étape les détails des activités de production. Au niveau planification on calcule précisement le meilleur plan de fabrication, compte tenu des demandes et de la capacité exacte de production. Cependant, on s'assure que ce plan est réalisable au niveau opérationnel. Pour fournir un plan de production réalisable, le niveau planification intègre des considérations d'ordonnancement sur un horizon plus long que l'horizon traditionel du niveau opérationnel.

La planification des opérations vise la répartition des ressources de l'entreprise en fonction de ses objectifs stratégiques, des contraintes existantes et de la demande prévue. A l'intérieur du système de pilotage, la planification des opérations concerne plutôt le moyen et le court terme puisque les capacités de production ne peuvent pas varier significativement au cours de la période de planification. Dans les processus de production, la planification est la réalisation d'un plan qui contient la préparation des matériaux, la sélection des opérations, leur séquencement, la sélection des machines, des outils et des autres ressources qui interviennent dans ces processus. Quand un produit est réalisé sur une ligne automatisée, le plan du processus contiendra les détails d'activités étape par étape. Le planificateur du processus automatisé obtient automatiquement les données d'entrée et produit un ensemble complet de plans qui seront utilisé dans la plannification du processus. A l'aide de l'ordinateur, des progrès sont faites à touts les niveaux. La complexité des systèmes de production modernes nécessite une utilisation très large de l'informatique pour l'optimisation ou l'aide à la décision.

Les objectifs de la planification des opérations au sein du pilotage consistent donc à déterminer les quantités de produits à réaliser à l'intérieur d'un horizon à moyen ou à court terme (planification des priorités) et à préciser les quantités de ressources à utiliser (la planification des capacités) afin de répondre le mieux possible aux objectifs opérationnels visés en matière de qualité, de volume, de lieu, de temps et de coûts [Nollet, 1994].

### 2.1.1.1. Les étapes de la planification

*1) La planification agrégée*

La première étape est **la planification agrégée** qui a pour objectif la détermination sous une unité de mesure commune des quantités globales de produits finis à réaliser au cours de la période constituant l'horizon de planification, et la détermination des volumes des principales ressources à utiliser selon divers modes disponibles, afin de satisfaire l'ensemble de la demande prévue pour la période. Cette étape porte sur le plus long horizon de planification utilisé en commande. Dans cette étape on trouve souvent des plans de production pour des périodes allant de six mois à deux ans. Un horizon de plus de deux ans ne concerne pas la commande, mais plutôt la conception des systèmes de production.

La planification du processus de production est la transformation des informations abstraites sur un produit dans des instructions de production pour ce produit. Le planificateur utilise les données obtenues dans la phase de conception du produit et les informations sur les ressources du système. La phase de planification doit assurer tous les détails pour les étapes de fabrication. Dans la documentation élaborée suite à cette phase on doit préciser le séquencement des opérations, les ressources, les matériaux, les outils nécessaires pour le développement du processus de production. Le niveau planification calcule précisément le meilleur plan de fabrication, compte tenu des demandes et de la capacité de la cellule ou d'atelier. Le résultat de cette étape est un document appelé le *plan général de production*.

*2) La planification détaillée*

La deuxième étape est **la planification détaillée** qui consiste dans la détermination des quantités de chaque type de produits finis à lancer en production à chaque période afin de satisfaire les demandes des clients en matière de volume et de temps, et de respecter les contraintes établies dans le plan de production. A cette étape, les demandes sont établies à partir de prévisions de la demande. L'horizon utilisé en planification détaillé est plus court, il ne devrait pas dépasser six mois, la durée la plus courante étant un trimestre divisé en semaines. Le document issu de cette étape s'appelle le programme directeur de production (PDP).

*3) La planification MRP*

**La planification des besoins matières** (*Material Requirements Planning* - MRP) est la troisième étape de la planification des opérations [Rembold, 1994]. Dans cette étape on détermine la date et les quantités de chaque matière première, pièce ou composant à commander ou à produire afin de pouvoir réaliser le programme directeur de production élaboré à l'étape précédente. Le document issu de cette étape s'appelle *le plan des besoins matières* (PBM). Le PBM sert maintenant à la planification autant qu'au contrôle. Le contrôle vise également à remédier aux retards et aux écarts qui risquent d'entraîner des modifications du PDP. Le PBM permet également de diriger les ressources. Le système s'étend ainsi à la planification des ressources de production (*Manufacturing Resources Planning* – MRP II). Une fois les quantités et les dates de fabrication et de commande connues, il est possible de passer à l'étape suivante.

*4) L'ordonnancement*

La dernière étape est **l'ordonnancement** dont l'objectif consiste à organiser dans le temps la réalisation de l'ensemble des tâches, compte tenu des contraintes temporelles et des contraintes portant sur l'utilisation et la disponibilité des ressources requises [Lopez, 2001].

L'ordonnancement est la répartition temporelle des tâches sur les machines, il est déterminé de façon à optimiser la performance du système de fabrication. On parle de problème d'ordonnancement lorsqu'on doit déterminer les dates de début et les dates de fin des tâches. Pour le cas où on cherche à établir un ordre relatif entre les tâches évitant les conflits pour l'utilisation des ressources on utilise le terme de *séquencement*.

Pour cette étape, l'horizon temporel est plus court que celui des étapes précédentes. On utilise des horizons allant d'une semaine à un mois. L'horizon peut être divisé en sous unités allant d'un jour à une semaine. Des *commandes* ou des *ordres de production* précises doivent être attribuées selon une séquence précise à chaque unité de ressource. Le document issu de cette étape est le *calendrier* ou *le programme de production*. La figure suivante présente les étapes de la planification des opérations dans un système de production.

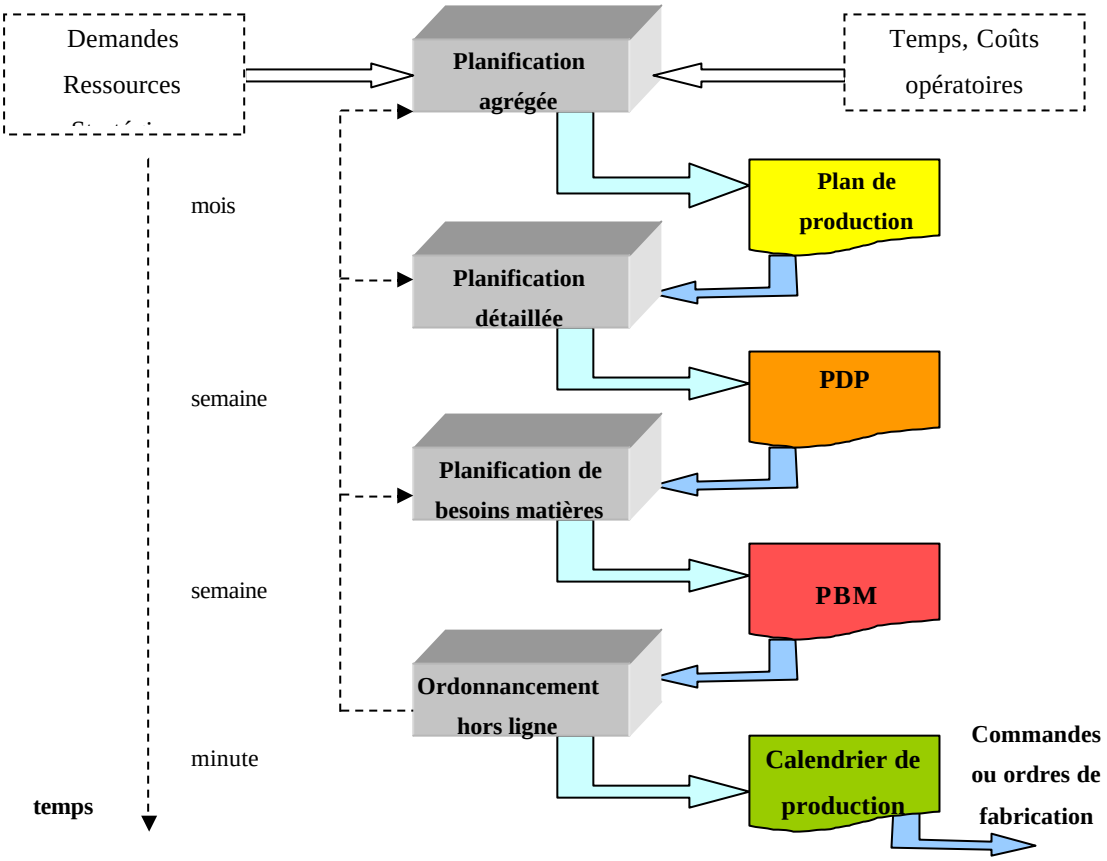



Fig. 10. Les étapes de planification

Notre travail se concentrera sur l'étape d'ordonnancement puisque l'ordonnancement est le lieu de l'interface entre l'élaboration globale de la commande et la partie opérationnelle du processus. Dans les paragraphes qui suivent nous allons décrire les éléments qui interviennent dans un problème d'ordonnancement.

## 2.2. La problématique de l'ordonnancement

L'ordonnancement est l'une des composantes clés du contrôle des activités de production. Son bon fonctionnement repose sur les informations adéquates provenant des modules de la phase de planification. La planification de production crée les ordres de fabrication qui déterminent ce qui doit être fait dans une période donnée (généralement la semaine) tantdis que l'ordonnancement consiste à prévoir l'enchaînement de toutes les opérations élémentaires nécessaires à la réalisation de ces ordres de fabrication sur les ressources de production en tenant compte des contraintes extérieures.

**Définition 2.1.**

*L'ordonnancement consiste à organiser dans le temps la réalisation d'un ensemble de tâches, compte tenu de contraintes temporelles et de contraintes portant sur l'utilisation et la disponibilité des ressources requises par les tâches. Un ordonnancement décrit l'ordre d'exécution des tâches et l'allocation des ressources au cours du temps afin de satisfaire un ou plusieurs critères d'optimisation* [Lopez, 2001].

### 2.2.1. Les éléments d'ordonnancement

D'après la définition citée ci-dessus, nous en déduisons qu'un problème d'ordonnancement est constitué principalement de quatre éléments: *les tâches, les ressources, les contraintes* et *les critères d'optimisation*.

#### 2.2.1.1. Les tâches

Les tâches sont toutes les opérations élémentaires de travail à effectuer pour la fabrication d'un produit.

*Une tâche* est un travail élémentaire dont la réalisation nécessite une certaine durée et un nombre de ressources. Une tâche $i$ est localisée dans le temps par une date de début $s_i$ (start time) ou de fin $c_i$ (completion time) dont la réalisation est caractérisée par une durée $p_i$. On a ainsi $c_i=s_i+p_i$. Certaines contraintes techniques ou économiques peuvent conduire à associer aux tâches des dates de début au plus tôt $r_i$ (ready time) ou des dates de fin au plus tard $d_i$ (due date).

#### 2.2.1.2. Les ressources

L'exécution des différentes tâches nécessite la mise en oeuvre d'un ensemble de moyens techniques et d'opérateurs humains. Cet ensemble représente donc les ressources indispensables à la réalisation des tâches durant les intervalles de disponibilité. *Une ressource* est un moyen, technique ou humain, dont la disponibilité limitée ou non est connue *a priori.* Une ressource est ainsi caractérisée par sa capacité supposée constante pendant le travail. Il y a plusieurs types de ressources. Si après son utilisation la ressource est à nouveau disponible avec la même capacité on parle d'une ressource renouvelable. Par contre, si après son utilisation la ressource est disponible avec une capacité inférieure ou nulle on parle de ressource consommable. Certains ressources sont capables de réaliser plusieurs tâches en parallèle, elles sont dites cumulatives (une station de travail composée de plusieurs machines est une ressource cumulative). Les ressources sont supposées disjonctives si elles ne permettent de réaliser qu'une seule opération à la fois (un ouvrier s'occupant de plusieurs machines est une ressource disjonctive).

### 2.2.1.3. Les contraintes

Les contraintes expriment des restrictions sur les valeurs des variables de décision. Leur prise en compte permet d'avoir un ordonnancement réalisable. Les contraintes qui sont liées directement au système de production et à ses performances sont les contraintes temporelles et les contraintes de ressources.

Les contraintes temporelles reflètent la non disponibilité continue des ressources et la nécessité de délivrer les produits en respectant les délais. C'est à dire que certaines opérations ne peuvent s'exécuter qu'après une date de début au plus tôt satisfaisant l'inégalité $c_i - p_i \geq r_i$ et ne finissent qu'avant une date de fin au plus tard ce qui revient à vérifier l'inégalité $s_i + p_i \leq d_i$. Les contraintes temporelles peuvent aussi définir des relations de précédence entre les opérations. Ce sont surtout des contraintes technologiques qui imposent un ordre d'exécution aux opérations. Le relation de précédence entre la tâche $i$ qui précède la tâche $j$ peut s'écrire sous la forme suivante: $s_i + p_i \leq s_j$.

Les caractéristiques des ressources peuvent induire des contraintes indiquant les conditions de passage des tâches. Dans le cas d'une ressource consommable, seules les tâches ayant une consommation inférieure ou égale à la capacité de la ressource à l'instant considéré peuvent être exécutées. Pour une ressource renouvelable ce type de contraintes exprime la limitation de sa capacité. Pour assurer le respect des contraintes de ressources on doit éviter le chevauchement temporel de certaines tâches. Ceci conduit à ordonner un nombre de paires des tâches ce qui revient à ajouter de nouvelles contraintes de succession à l'ensemble initial des contraintes. Ce type d'action est appelé *séquencement*. Le problème de séquencement est induit par le problème d'ordonnancement initial.

### 2.2.1.4. Les critères d'optimisation

L'objectif de l'ordonnancement est d'optimiser (maximiser ou minimiser) une fonction d'évaluation en respectant un certain nombre de contraintes. Les critères d'optimisation dans l'ordonnancement classique sont le coût, le délai de livraison et la qualité de processus.

*Le coût* des ressources est un critère assez rare parce que le coût machine est invariant. Ce coût est pris en compte dans la phase de conception de la ligne. Le coût de production est souvent utilisé comme critère d'optimisation (minimisation). Les arrêts et les temps d'inactivité augmentent les coûts de production. L'objectif de l'ordonnancement est ainsi de minimiser le temps mort entre les opérations et donc d'assurer une activité continue pour chaque poste de travail ou chaque machine.

*Le délai* évalue en termes de temps les performances d'un système. Respecter le délai consiste à définir un ordre de fabrication durant une période donnée.

*La qualité* est rarement associée à l'évaluation d'un ordonnancement. Toutefois, la fluidité d'un flux de fabrication est associée à un facteur de qualité. La qualité d'un processus de fabrication est obtenue en minimisant le nombre des produits inachevés pendant une période de production. Malheureusement, ce critère est rarement utilisé dans l'ordonnancement théorique.

### 2.2.2. Les étapes d'ordonnancement

L'objectif du problème d'ordonnancement est de fixer la date de début $s_i$ pour chaque tâche *i*. Pour cela il faut déterminer l'ordre de passage de l'ensemble des travaux ou séquence sur chaque machine. A partir de ces séquences plusieurs ordonnancements admissibles (localisant de manière absolue les opérations dans le temps) peuvent être obtenus selon le critère à optimiser. L'ordonnancement comme processus de décision a plusieurs étapes. Les plus importantes sont l'affectation des tâches au postes et leur séquencement.

L'étape de **l'affectation des tâches** consiste à répartir les tâches aux divers postes de travail en fonction de ressources disponibles et des précédences entre les tâches. Dans cette étape l'affectation des tâches aux ressources n'est pas fixée a priori. A chaque tâche *i* est associé un ensemble de ressources capable de la réaliser. Pour chaque ressource la durée de réalisation $p_i$ de la tâche est déterminée. Cette durée peut être variable ou fixe. On distingue les problèmes en fonction de caractéristiques des ressources: lorsque les ressources sont identiques, la durée d'une tâche ne varie pas selon la ressource à laquelle elle est affectée; lorsque les ressources sont uniformes, la durée varie de manière proportionnelle selon la ressource choisie; lorsque les ressources sont indépendantes, la durée d'une tâche est quelconque en fonction de la ressource utilisée pour sa réalisation.

L'étape de **séquencement** est **la détermination d'un ordre de passage** (ou séquence) qui consiste à établir un ordre dans lequel les différentes tâches seront exécutées sur chaque poste de travail. Pour ce faire, on doit connaître la gamme complète et les durées opératoires pour chaque tâche. L'évaluation de l'ordre de passage repose sur plusieurs critères: la minimisation des coûts de travail et des stocks ou la maximisation des taux d'utilisation des postes de travail. Souvent il est préférable que cette étape définisse un ordre de passage qui donne des résultats satisfaisants dans des conditions réalistes, plutôt qu'un ordre qui donne des résultats théoriques optimaux, mais trop éloignés de la réalité.

Une fois réalisées l'affectation des tâches à chaque poste et la détermination du séquencement, on peut alors passer à la troisième étape qui consiste à **établir le calendrier de fabrication**. Ce dernier donne une date précise pour le lancement de chaque opération à chaque poste de travail. Toute variation dans la gamme et la durée des opérations modifie ce calendrier et peut même conduire à une révision des deux premières étapes.

La quatrième étape de l'ordonnancement est **le lancement de la production**, soit l'exécution du calendrier.

La phase de **contrôle de l'ordonnancement** permet le suivi de l'exécution du travail par le biais de la collecte de données sur les commandes en cours. Tout retard dans la production est enregistré dans un rapport qui sera utilisé dans la phase de l'ordonnancement pour revoir les priorités de production.

### 2.2.3. Les approches d'ordonnancement

La résolution d'un problème d'ordonnancement doit concilier deux objectifs: *l'aspect statique* – qui consiste à générer un plan de réalisation des travaux sur la base de données prévisionnelles et *l'aspect dynamique* qui consiste à prendre des décisions en temps réel compte tenu de l'état des ressources et de l'avancement dans le temps des différentes tâches.

Ainsi, dans la littérature on trouve deux approches systémiques du problème de l'ordonnancement: une **approche statique** et une **approche dynamique**.

#### 2.2.3.1. L'ordonnancement hors ligne

Dans une *approche statique*, qui s'appelle aussi *ordonnancement prévisionnel* ou *ordonnancement hors ligne*, l'ordonnancement se fait une seule fois sur l'horizon considéré (jour, semaine) pour toutes les commandes reçues. Une nouvelle commande nécessitera le changement de l'ordonnancement d'où la nécessité d'une prévision juste dans la préparation du PDP.

L'ordonnancement prévisionnel groupe les dates de début et de fin des différentes opérations réalisées sur les machines sur l'horizon considéré. Il vise à prévoir les modalités d'exécution d'un ensemble de tâches par un ensemble de ressources sur un horizon fini ou infini effectuant des hypothèses préalables sur l'évolution du système au cours du temps.

Il y a deux approches : la première consiste à construire par une génération progressive une séquence initiale qui satisfait toutes les contraintes sauf éventuellement la date de livraison, et la seconde étape qui consiste justement à modifier la séquence initiale pour chercher à satisfaire la demande et la date de livraison. L'ordonnancement hors ligne définit l'affectation des tâches sur les machines. Pour satisfaire les ordres de fabrication, l'opération candidate à l'affectation est la première opération de la gamme dont l'exécution n'est pas commencée.

L'approche décentralisé prévisionnelle consiste à définir une politique de gestion des files d'attente devant chaque ressource qui est validée grâce a une simulation de l'écoulement des travaux dans l'atelier. Cette simulation prend en compte aussi les aspects dynamiques du problème et permet de comparer les performances obtenues par différentes politiques afin d'en retenir une. C'est celle qui sera effectivement appliquée pour construire en temps réel l'ordonnancement de l'atelier et l'approche par simulation.

2.2.3.2. L'ordonnancement en temps réel

Dans un *approche dynamique*, appelé aussi *ordonnancement en ligne*, on doit prendre en considération toutes les commandes quel que soit le moment de leur arrivée. Dans ce cas, l'ordonnancement devient une activité très complexe faisant appel aux techniques mathématiques de recherche opérationnelle et à la simulation.

On parle d'ordonnancement dynamique ou "en temps réel" quand les décisions d'ordonnancement des opérations sur une machine sont prises une par une chaque fois qu'une machine se libère ou quand une opération devient disponible. Ainsi, l'ordonnancement est effectué petit à petit au cours du temps. L'ordonnancement en ligne est très difficile à accomplir car la décision d'affectation d'une tâche sur une machine doit être prise au moment quand la machine se libère. Dans ce cas l'ordonnancement vise à fournir un ensemble de solutions compatibles avec les contraintes en temps réel. Une vision dynamique de l'ordonnancement s'impose afin de pouvoir réagir rapidement aux différents aléas: travaux imprévus à réaliser, panne de ressources, absence de composants, défauts d'approvisionnement. Dans le cas du désassemblage le problème d'ordonnancement est plus compliqué du fait de l'impossibilité d'affectation des certaines opérations de démontage sur certaines stations de travail et de leurs contraintes de précédences (voir le Chapitre suivant). Cette caractéristique du problème d'ordonnancement permet de le considérer comme un *problème de prise de décision en temps réel*, afin, connaissant l'état réel de la ligne et des ressources, de prendre en temps réel des décisions qui vont assurer un bon développement du plan de fabrication. Ainsi, un système d'aide à la décision est introduit dans la boucle de commande.

La figure 11 présente le dialogue entre le système d'aide à la décision (SAD) et le système de contrôle et commande (SCC).

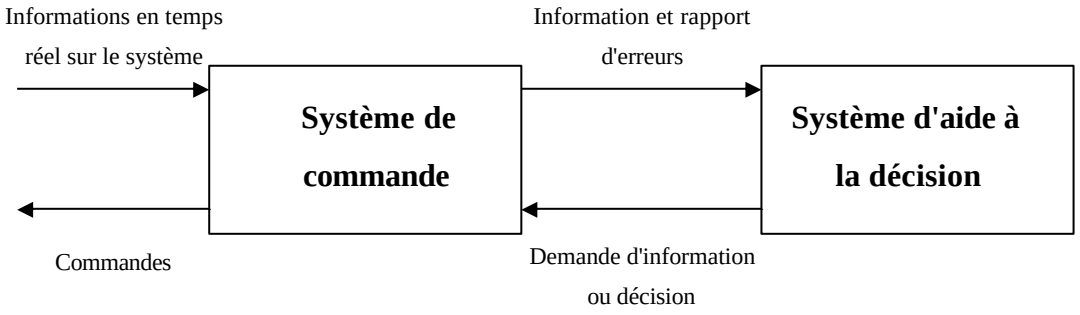



Fig. 11 Le flux d'information entre le SAD et SCC

Le rôle du système d'aide à la décision est d'envoyer des messages vers le système de commande du processus. Ce message est soit une demande d'information soit une décision. Pour chaque requête, le système de contrôle et commande identifie les entités concernées et renvoie des informations sur ces entités. Si le message est une décision, le SCC vérifie la validité de cette décision en testant l'état réel du système de production. Si le test est négatif, une information de requête non valide est envoyée au centre de

décision avec les explications de l'erreur. Si la requête est valide, l'information est mise à jour et les temps opératoires sont recalculés en fonction du moment présent.

Quatre sortes de décisions sont considérées pour l'aide à la décision en temps réel: la décision de *commencer* l'exécution d'une opération, la décision de *reprendre* une opération (la fin d'une panne), la décision d'*interrompre* l'exécution d'une opération et la décision de *changer l'affectation* d'une opération. Si la dernière décision se prend en temps réel on parle d'*affectation en temps réel*. Un modèle dynamique du système de production est nécessaire pour représenter l'enchaînement des états en fonction des décisions et des événements qui peuvent survenir.

### 2.2.4. La complexité des problèmes d'ordonnancement

Les problèmes d'ordonnancement sont d'habitude des problèmes combinatoires pour lesquels il existe des outils généraux de résolution (tels que la programmation dynamique) à la condition que le critère d'optimisation présente des propriétés particulières, par exemple une forme additive. La complexité d'un algorithme d'ordonnancement est jugée par rapport au temps de calcul nécessaire pour trouver la solution [Carlier, 1991].

**Définition 2.2.**
On appelle *complexité en temps d'un algorithme* la fonction $f(n)$ qui représente le nombre maximum d'opérations élémentaires effectuées pour résoudre un problème de n variables.

**Définition 2.3.**
On appelle ordre de complexité d'un algorithme un ensemble des fonctions, noté $O(f(n))$, qui remplit la condition suivante : $\forall g(n) \in O(f(n)) \Rightarrow \lim_{n\to\infty} \frac{g(n)}{f(n)} = 1$

Par rapport au type de la complexité on peut avoir des algorithmes polynomiaux, exponentiels, logarithmiques.

Exemples d'ordres de complèxité :

$O(1)$ - Algorithme constant

$O(\log n)$ - Algorithme logarithmique

$O(n)$ - Algorithme linéaire

$O(n!)$ - Algorithme factoriel

$O(n^k)$ - Algorithme polynomial (k entier)

$O(k^n)$ - Algorithme exponentiel (k entier)

**Définition 2.4.**
Un algorithme est dit polynomial si sa fonction de complexité $f(n) \in O(p(n))$ où $p$ est un polynôme en $n$.

**Définition 2.5.**
Un problème est NP-complet s'il n'existe pas un algorithme polynomial pour le résoudre.

**Définition 2.6.**
Un problème d'optimisation est dit NP-difficile s'il appartient à la classe des problèmes NP-complets.

Dans notre travail le problème d'ordonnancement est de type flow-shop (cf. Chapitre 3). L'ordre du problème de flow-shop à 2 machines est $O(n \cdot \log n)$ ou $n$ est le nombre de tâches à faire sur les 2 machines. Le problème de flow-shop devient NP-difficile au-delà de 3 machines [Lopez, 200].

La complexité du problème d'ordonnancement peut être résolue à l'aide des logiciels existants ou par des méthodes heuristiques [Kuo, 2000].

Depuis quelques années des méthodes heuristiques en conjonction avec la simulation sont utilisées pour obtenir un ordonnancement prévisionnel. Le rôle de la simulation est d'obtenir l'affectation des tâches sur un horizon temporel, qui s'appelle *horizon d'ordonnancement*. Le problème d'ordonnancement est généralement multicritère. Pour chaque valeur d'une variable de décision il y a une affectation possible des tâches sur les postes de travail. Dans l'approche heuristique plusieurs simulations sont effectuées avec des valeurs de décision différentes. Il y a plusieurs règles heuristiques qui peuvent être utilisées dans l'ordonnancement : premier entré - premier sorti, la règle du pourcentage optimal, la règle de Carroll, la règle du temps d'attente [Filip, 2005]. De plus, les méthodes heuristiques d'ordonnancement peuvent être implémentées à l'aide de logiciels simples pour la simulation (par exemple la méthode Monte Carlo implémentée à l'aide d'Excel).

## 2.3. L'activité de pilotage

### 2.3.1. Définitions

Il n'y a pas une définition unifiée du mot *pilotage* dans la littérature, mais on peut mentionner quelques fonctions et définitions de l'activité de pilotage d'un système de production. Le mot *pilotage* est choisi volontairement pour souligner l'interactivité par opposition au mot traditionnel de suivi de production, qui semble ne représenter qu'une course après la réalité sans intervention sur le processus.

Plusieurs définitions ont été utilisées pour *l'activité de pilotage* [Borne, 1992]

**Définition 2.7.a.**
Le pilotage est le chaînon entre la GPAO et la commande des moyens de fabrication ;

**Définition 2.7.b**
L'activité de pilotage est l'ordonnancement en temps réel des tâches

**Définition 2.7.c**
L'activité de pilotage cherche à optimiser la relation entre les hommes, les machines, les stocks et les mouvements physiques ;

**Définition 2.7.d**
Le pilotage comme processus doit gérer la coordination de l'ensemble des opérations.

**Définition 2.7.e**
Le pilotage à pour le but l'équilibrage de la charges des postes de travail et l'optimisation des gammes opératoires.

**Définition 2.8.** [Yin, 1995]
*Le pilotage consiste à commander et contrôler la partie opérative du système pour que le déroulement de la production soit conforme à l'objectif de production tel qu'il a été défini par les niveaux supérieurs.*

Les ressources pilotées sont le personnel (emploi, transferts entre centres de charges), l'outillage (équipements pour préparer les opérations), les machines (les capacités disponibles pour absorber la charge), ou les matériaux (les stocks).

Le pilotage peut agir au niveau de l'atelier, de la cellule ou du poste de travail. Piloter une cellule de production, c'est en fait accomplir un certain nombre de fonctionnalités comme : l'ordonnancement en temps réel des tâches, la gestion des dysfonctionnements au niveau de la cellule et au niveau des postes, la prise en compte des consignes du niveau supérieur, la gestion des communications entre les différents composants de la cellule. Le pilotage d'un poste doit gérer la conduite de la machine de telle façon qu'elle traite les produits d'une manière autonome dans un minimum de temps.

### 2.3.2. Les fonctions du pilotage

B. Yin dans sa thèse [Yin, 1995] a spécifié les *fonctions* d'un système de pilotage: l'ordonnancement à court terme, l'ordonnancement en temps réel, la commande, et la surveillance des tâches. Le but de l'ordonnancement à court terme est de déterminer l'affectation des tâches, c'est-à-dire, de déterminer l'ensemble des opérations à exécuter sur chaque poste. C'est un niveau décisionnel puisque il faut prendre une décision de manière à équilibrer les charges à distribuer aux postes.

L'ordonnancement en temps réel consiste en un ensemble de décisions de lancements de fabrication des produits et de séquencement d'exécution des opérations au niveau de postes. Cette fois la décision doit être prise en temps réel. Le niveau commande doit gérer les informations reçues et maîtriser la conduite des robots, outils, ou convoyeurs.

Le niveau surveillance recueille des informations sur l'exécution des décisions en temps réel et surveille le déroulement de la production.

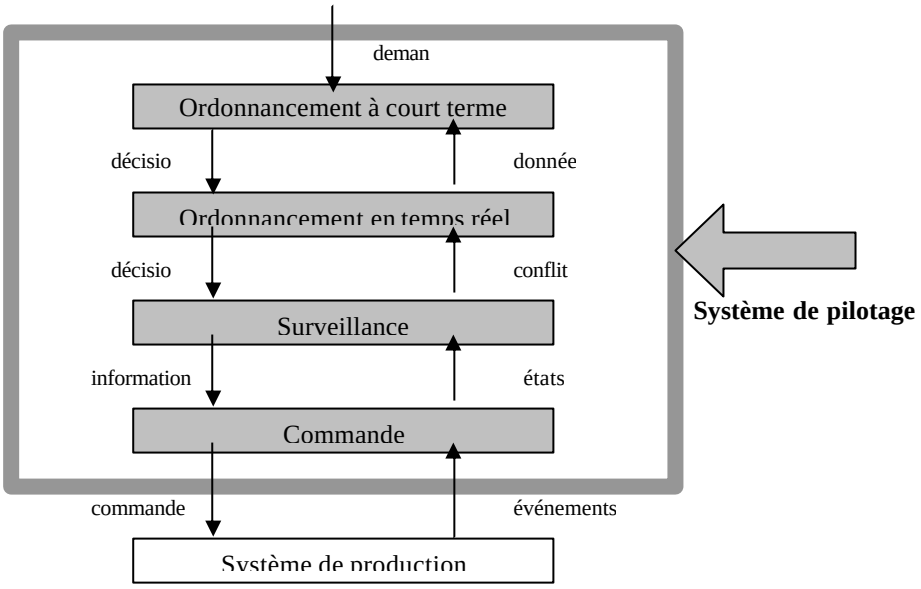



Fig. 12. Les fonctions du système de pilotage (selon Yin)

Ces quatre fonctions de pilotage concourent à la flexibilité et à la réactivité du système de production. La flexibilité caractérise l'adaptation du système aux tâches demandées, et la réactivité donne la promptitude à réagir face à une modification prévue ou non de l'environnement. Ainsi, on considère que les adaptations matérielles et décisionnelles au changement correspondent à la flexibilité du système de production et l'adaptation temporelle du système de pilotage correspond à la réactivité du système de production. Comme nous l'avons souligné précédemment, une des fonctions principales du pilotage est *l'équilibrage des tâches sur les postes de travail*, de telle sorte que les temps opératoires sur ces postes soient aussi proches que possible. Dans le paragraphe suivant nous reviendrons sur l'équilibrage.

Les critères importants qui doivent être pris en considération pour le pilotage sont l'abstraction et la modularité qui permettent une facilité d'intégration et l'ajout de nouvelles fonctionnalités. L'originalité du développement d'un système de pilotage tient dans le choix de l'architecture informatique matérielle et logicielle pour les communications et le traitement des informations. Le système informatique de pilotage doit gérer la circulation des pièces et des outils, détecter les anomalies de fonctionnement, assurer la sécurité, et optimiser le chargement des machines. Les systèmes de pilotage sont souvent techniquement moins difficiles à mettre en œuvre que les langages informatiques qui les commandent.

Au niveau des postes, l'ordinateur de commande est l'élément qui pilote les équipements (poste entièrement automatique). Il doit assurer trois fonctions principales :

- Fonction de commande : dans un processus avec des événements discrets en temps (comme le désassemblage) l'ensemble d'éléments discrets est commandé par des variables booléennes ;

- Fonction programme : assure les programmes de modélisation qui décrivent la suite des opérations à effectuer dans le but de transformer le produit et les programmes d'application qui correspondent aux fonctions de commande du processus;
- Fonction interface : assure le dialogue entre l'utilisateur, le système superviseur et le système de commande.

### 2.3.3 Le pilotage du désassemblage

Dans ce travail nous avons consideré que la commande des systèmes de désassemblage concerne *l'ordonnancement d'opérations en ligne*. Les opérations de désassemblage doivent être affectées aux postes de telle manière qu'il n'y a pas de temps morts ou de machines trop chargées. C'est à dire que les temps de travail pour chaque poste doivent être approximativement égaux (en manuel essentiellement).

Mais dans le cas du désassemblage ce problème devient plus difficile à cause du mauvais état des produits. On ne sait pas à l'avance si les composants sont à leurs place ou s'ils peuvent être extraits. Si un sous-ensemble ne peut être désassemblé on se trouve dans la situation de refaire l'ordonnancement des opérations pour extraire un autre composant ou pire, d'arrêter le processus de désassemblage. Cette situation n'est pas acceptable donc on doit prendre en considération un pilotage réactif de la ligne.

Le pilotage réactif suppose une affectation en temps réel des opérations aux ressources disponibles en fonction de l'état du produit et du système en général. Dans ce cas l'équilibrage de la ligne doit être fait aussi en temps réel.
Pour faire face aux incertitudes il faut tenir compte des dégradations possibles lors de la planification du désassemblage. L'état réel du produit à l'entrée d'une chaîne de désassemblage est souvent inconnu. Le produit peut être modifié par des procédures de réparation, remplacement de différents sous-ensembles, ou par les dégradations dues à son usage.

Pour la conduite des processus de désassemblage il faut trouver une méthode basée sur une approche dynamique, permettant la réaction en temps réel aux événements imprévus comme par exemple l'échec des opérations de démontage. Dans le même temps, l'équilibrage des charges des postes de travail doit être assuré dynamiquement pendant le processus de désassemblage en fonction de l'état réel du système.

Les deux problèmes, d'ordonnancement et d'équilibrage de la ligne de désassemblage, sont trop importants pour êtres résumés ici. On leur a dédié un chapitre spécial dans notre travail en tenant compte qu'il n'y a pas une approche théorique systématisée à l'heure actuelle.

### 2.3.4. Conception conjointe commande et ordonnancement

Les tâches de commande sont soumises à des incertitudes temporelles. Conjointement à l'algorithme de commande les paramètres d'ordonnancement d'un programme de commande doivent être adaptés ou optimisés en vue d'améliorer les performances d'un

contrôleur, en particulier sa robustesse temporelle. D'autre part les techniques de commande en boucle fermée peuvent être appliquées à l'ordonnanceur temps réel lui même, le rendant ainsi adaptatif et robuste face aux incertitudes temporelles d'exécution. La détermination de la durée d'exécution des tâches est un problème difficile.

Une première approche consiste à calculer, hors-ligne, une affectation des paramètres d'ordonnancement maximisant la performance de commande du système sous contrainte d'ordonnançabilité des tâches. Le point de départ est bien entendu d'obtenir un modèle de performance fonction des paramètres d'implémentation. Une autre possibilité consiste à formuler le problème d'implémentation de commande sous forme d'un critère de qualité de service rendant compte, par exemple, de la relation performance/période de chaque contrôleur dans le système.

Les approches précédentes supposent que l'ordonnancement est prédéfini. Mais dans le désassemblage les durées des opérations ne sont pas connues a priori. Le manque d'un composant ou les défauts d'un autre influencent ces durées. La variété des caractéristiques temporelles et l'incertitude liée aux valeurs, connues uniquement à l'exécution de ces activités, font qu'il est nécessaire de complèter un ordonnancement statique par un ordonnacement en temps réel..

La solution est d'adapter en temps réel l'ordonnancement en fonction du succès ou de l'échec des opérations de désassemblage. Dans ce cas il s'agit d'une commande réactive aux événements imprévus du système. Le but de notre travail est de trouver des algorithmes appropriés pour l'ordonnancement du désassemblage et de réaliser une commande réactive du processus.

## 2.4. Etat de l'art

### 2.4.1. Architectures de conduite

En général, la conduite des systèmes flexibles de production fait appel à la modélisation du système à piloter. Vu la complexité de cette conduite, il est nécessaire de décomposer la modélisation du système sur les différents horizons temporels: la production à long terme, à moyen terme ou à court terme. Un autre manière de modéliser les systèmes de conduite est de tenir compte de type du système de production : job shop ou flow-shop.

Plusieurs architectures de conduite ont été identifiés par [Adamou, 1997]. Une première architecture est décrite pour les systèmes orientés job-shop. Un système de production est dit de type job-shop s'il est structuré suivant un regroupement des tâches avec des technologies semblables et suivant l'agencement des opérations relatives à une tâche. Cinq niveaux pour la conduite du système ont été proposés au LAAS (Toulouse) : planification, ordonnancement, supervision, coordination, commande locale, et le niveau du système physique.

La conduite des niveaux planification et ordonnancement s'effectue hors ligne. La supervision assure la politique de gestion des tâches. Cette architecture de conduite ne

prend pas en considération ni les aléas de production, ni le suivi de la production. Mais pour les systèmes de désassemblage, le suivi des opérations est essentiel, car il permet au système de commande de réagir en temps réel aux événements imprévus (comme l'impossibilité de démonter tel composant).

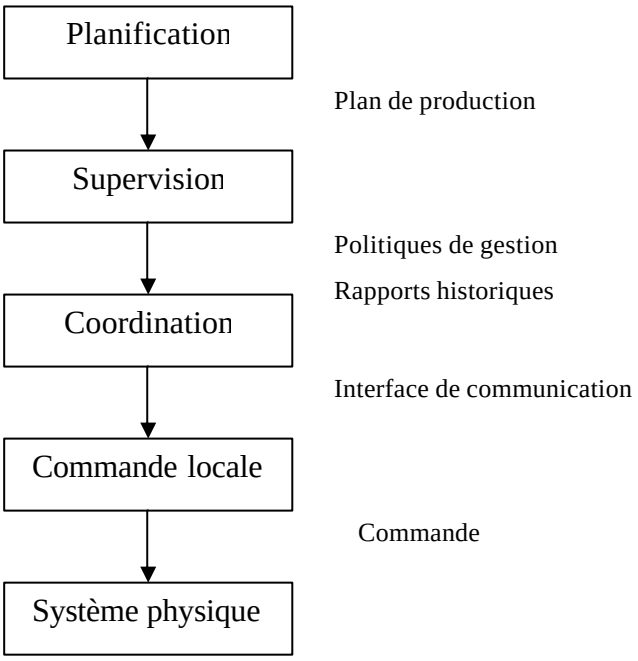



Fig. 13 Architecture de conduite proposée au LAAS

Une architecture plus intéressante a été proposée pour un système de conduite à Valenciennes. Elle est structurée sur six niveaux: planification, programmation, ordonnancement, conduite, commande, exécution (Figure 14). Au niveau planification il y a une vision à long terme sur la production: plan directeur, objectifs financiers et de fabrication. On peut dire qu'au niveau programmation on établit une prévision de la production à moyen terme (une semaine). Cette prévision caractérise les besoins et les stocks en cours. Le niveau de conduite est chargé de réaliser la production prévue par le niveau ordonnancement. Le niveau commande joue un rôle d'interface entre le niveau conduite et les systèmes physiques. Un protocole de suivi de production du plus bas niveau au plus haut assure la synchronisation entre les différents niveaux de décision.

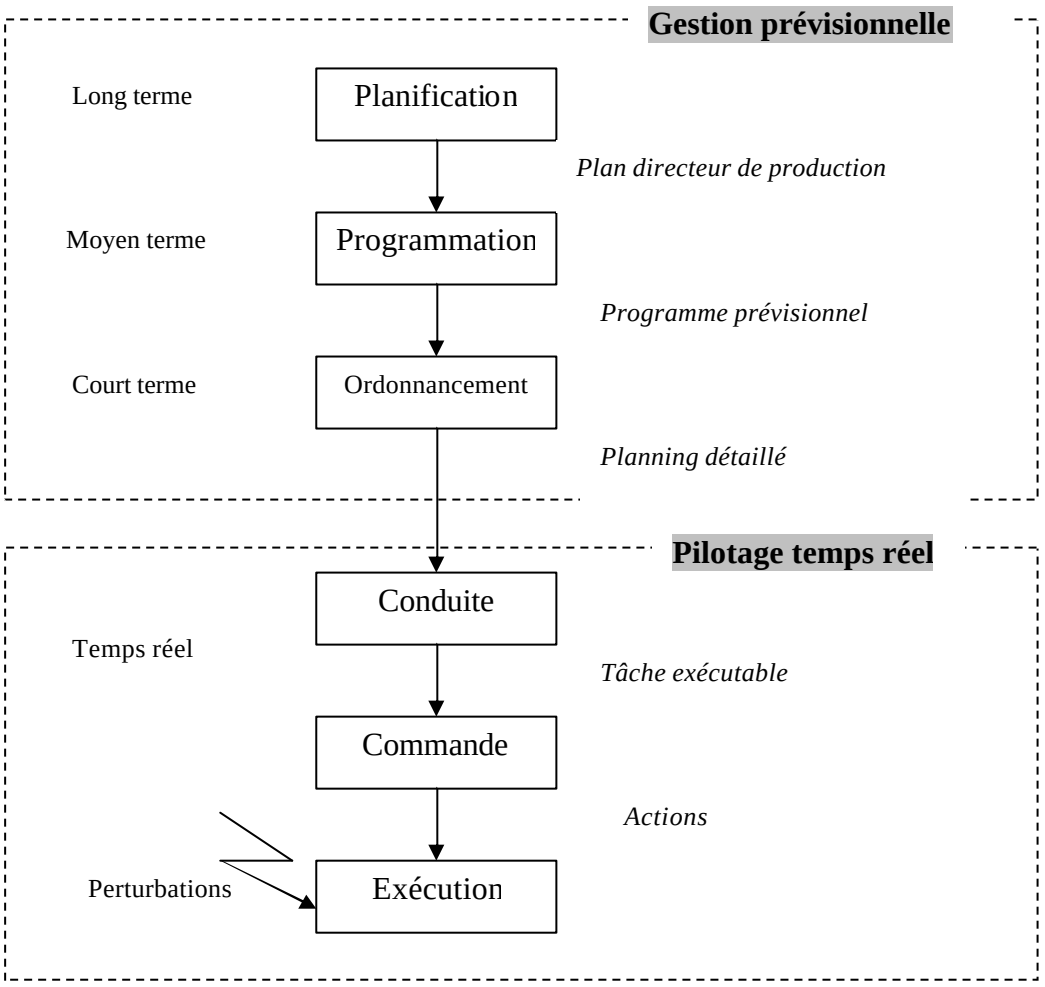



Fig. 14. Architecture de conduite (Valenciennes)

Le système de production de type flow-shop est caractérisé par le regroupement du flux matériel et l'agencement des opérations relatives à ce flux. Dans ce cas une structure différente est proposée dans [Adamou, 1997]. Il s'agit de la conduite d'une cellule d'assemblage dénommée DIAC (Delft Intelligent Assembly Cell) (Figure 15).

La conduite est subdivisée en quatre niveaux de décision organisés selon la composition du flux de production: le niveau lot, le niveau produit, le niveau composant, le niveau primitive. Au premier niveau s'effectue l'analyse de la faisabilité au sens de l'assemblage d'un lot de n produits. Au deuxième niveau, niveau produit, les séquences d'assemblage sont établies. Au troisième niveau la planification de la séquence d'assemblage des composants est faite. Chacun de ces niveaux a sa propre commande qui supervise son fonctionnement. Le quatrième niveau, dit primitive donne les primitives virtuelles des actionneurs et des capteurs physiques. Ces primitives constituent une bibliothèque d'objets constituant les modèles de ces actionneurs. La structure distribuée de cette architecture fait d'elle une structure coordonnée car les décisions sont bien distribuées entre les niveaux.

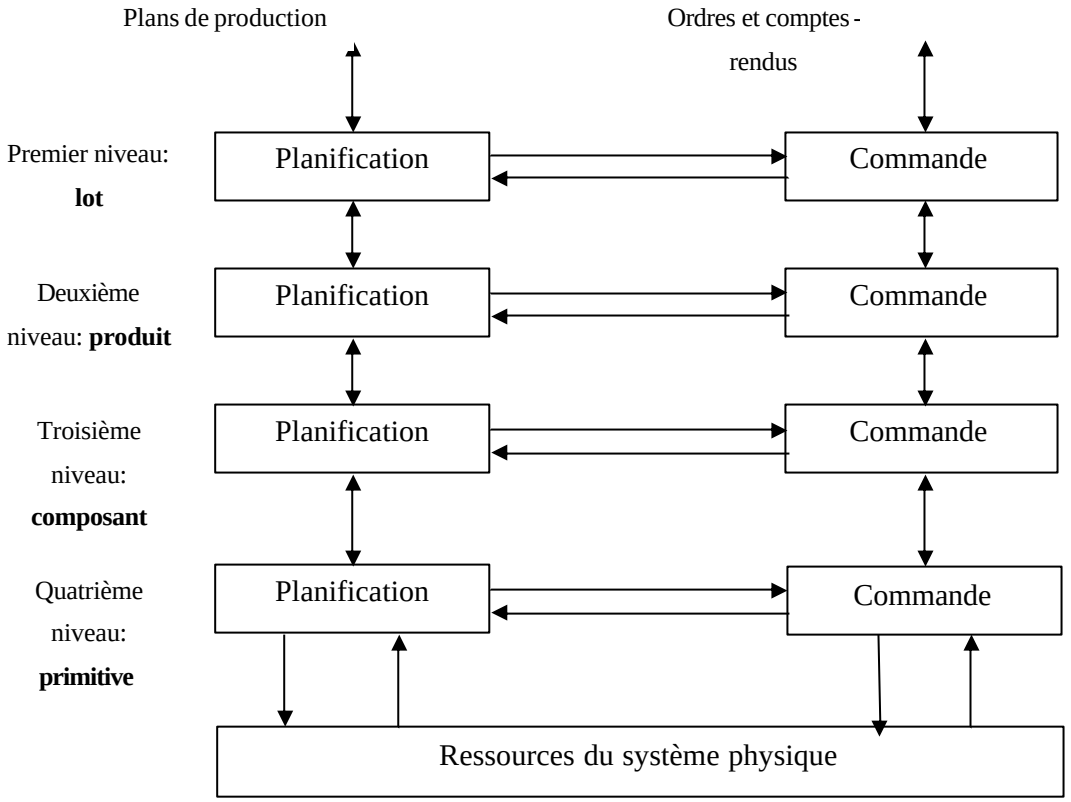



Fig. 15. L'architecture de conduite pour la cellule DIAC

Au Département d'Ingénierie Electrique de Zaragoza, Espagne une architecture qui combine les modèle de réseaux de Petri et le formalisme d'objet est implémentée pour la conduite d'un système de production. L'architecture est développée autour d'une base de connaissances qui se caractérise par une approche objet pour la représentation des entités et de leurs liens. Un dispatcheur a le rôle d'analyse et de prise de la décision pour ce qui concerne l'évolution de production. Cette architecture donne une bonne flexibilité au système de conduite mais elle ne satisfait pas les besoins d'un système flexible d'assemblage dans lequel le nombre de niveaux de prise de décision est supérieur à deux (figure 16).

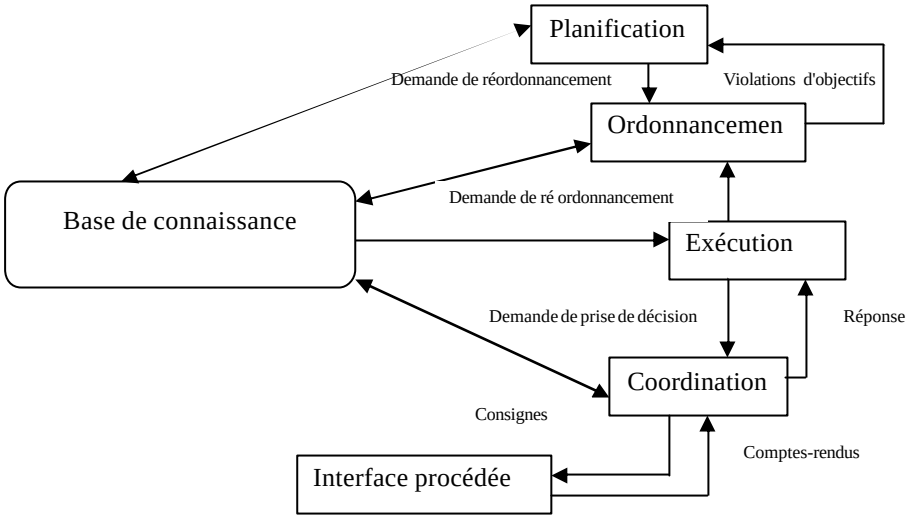
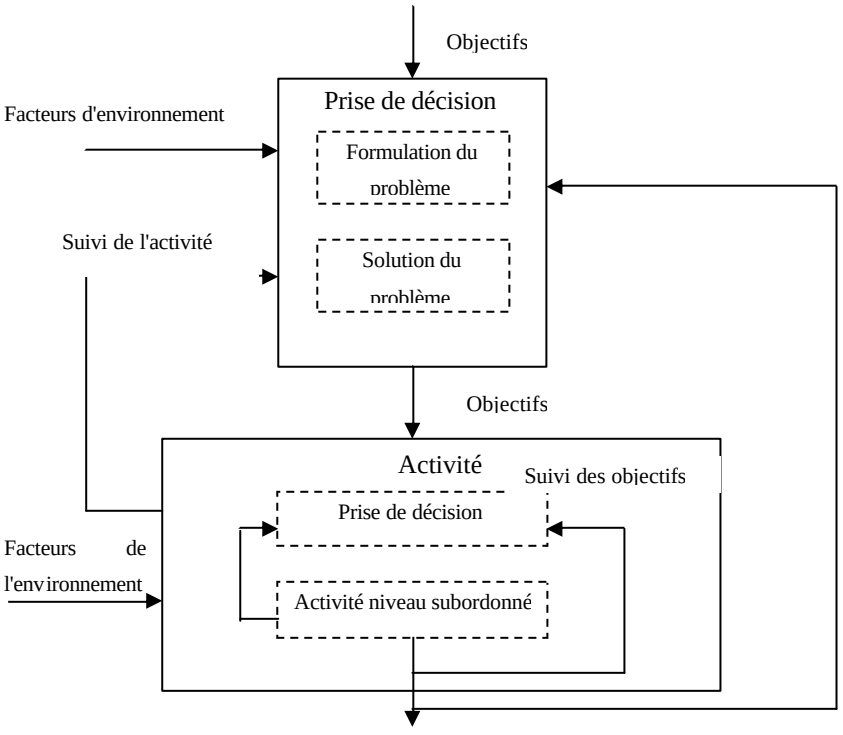



Fig. 16. Architecture développée autour d'une base de connaisance

Au département d'Information et Sciences de décision de l'Université de Virginia USA un modèle dynamique basé sur la hiérarchisation du système de production a été proposé (fig. 17).




.

Fig 17. Modèle dynamique de conduite

Ce modèle est caractérisée par deux activités en parallèle: une de planification qui établit les objectifs de production à attendre et une opérationnelle qui exécute ces objectifs. Une prise de décision relative à un problème se caractérise par une boucle de

conduite comprenant deux contre réactions ayant pour fonction de transférer l'activité qui lui est subordonnée.

Au LAB Yin [Yin, 1995] s'est inspirée des résultats des travaux d'analyse relatifs à la conception des systèmes flexibles d'assemblage pour proposer une approche de pilotage réactif basée sur un ordonnancement dynamique. Le niveau ordonnancement en temps réel prends deux types de décisions: le lancement des produits et le séquencement d'exécution des opérations au niveau de postes. Yin génère une réaffectation de ressources après une optimisation des coûts de production. La méthode est testée sur une ligne circulaire d'assemblage.

On a vu que chaque type d'architecture pour la conduite d'un système de production a un modèle spécifique en fonction du problème et du type du système soumis à l'étude. L'architecture *flow-shop* est très indiquée pour les systèmes d'assemblage, mais ne satisfait que partiellement une conduite réactive du système.

Compte-tenu de la grande incertitude sur l'état des produits à désassembler les approches générales relatives aux systèmes de type flow-shop doivent être repenser.

### 2.4.2. Etat de l'art de la planification du désassemblage

Dans le processus de planification de la production on doit décider quels produits sont nécessaires et quand. Dans la planification du désassemblage on cherche à établir les types des produits à désassembler pour satisfaire la demande. Le problème est donc la grande variété de produits qui doivent être désassemblés et leurs états incertains en fin de vie. Les caractéristiques géométriques et fonctionnelles des produits changent pendant la période d'utilisation. Ces circonstances demandent une nouvelle approche pour la planification et la commande des systèmes de désassemblage où l'information est parfois floue.

Les études dans le domaine du désassemblage portent dans le contexte de l'écologie industrielle ou de la conception pour le recyclage. Le processus de désassemblage est une partie de la production inverse (*reverse production*) qui intègre aussi des aspects de séparation et réparation de composants ou de recyclage. Viennent ensuite les niveaux de la planification et de l'ordonnancement prévisionnel du désassemblage. Au niveau opérationnel on est concerné par l'ordonnancement en temps réel et par le pilotage des systèmes de désassemblage.

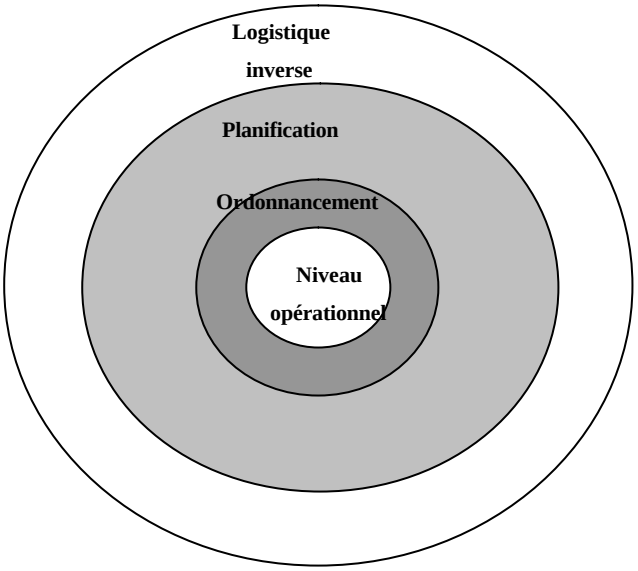



Fig. 18. Les domaines des études dans le désassemblage

Dans la littérature on ne trouve pas beaucoup des travaux dans les domaines de la planification et de l'ordonnancement du désassemblage.

Au début du processus de désassemblage l'information est incomplète. Avec l'avancement du désassemblage des nouvelles informations arrivent. Des opérations sont ajoutées ou supprimées. Les temps opératoires deviennent différents des temps prévus. Ainsi, le système de planification et de contrôle doit reconnaître les différences entre le processus planifié et le processus réel et il doit réagir immédiatement.

Plusieurs méthodes ont été proposées pour traiter ces difficultés [Wiendhal, 1999] : l'utilisation des outils pour la reconnaissance en ligne, l'utilisation des données floues, l'adaptation de la structure du système de commande à la fluctuation de la demande et aux aléas, l'intégration de la capacité de décision des opérateurs dans le système, l'implémentation des algorithmes d'optimisation robustes, adaptables au changement d'information.

Les approches utilisées pour étudier la planification du désassemblage ont été developpées sur plusieurs niveaux [Wiendhal, 1999] :

- *La planification des capacités* qui établit la quantité de produits démontables sur un horizon d'un ou plusieurs mois et les ressources nécessaires pour accomplir ces tâches;
- *La planification des charges* qui donne les quantités de produit à traiter période par période;
- *La planification du processus* lui même qui donne en principe la stratégie de désassemblage et les gammes opératoires;

Wiendahl et Selinger en [Wiendhal, 1999] ont bien relevé la liaison entre la planification des tâches et le séquencement des opérations de désassemblage. Dans leur travail la spécificité et la difficulté du désassemblage sont bien mises en évidence. Les auteurs proposent un système de planification et de contrôle hiérarchisé qui peut reconnaître les écarts entre le plan prévisionnel et l'activité réelle. A la fin de chaque opération l'état de l'ensemble obtenu est validé par un ouvrier ou par un système de capteurs. Le plan du désassemblage est trouvé à partir de la détermination de l'état du produit.

O'Shea et Kabernick [O'Shea, 1998] ont proposé un état de l'art de la planification du désassemblage. Ils font aussi une étude du séquencement des opérations de désassemblage pour obtenir la meilleure gamme. On trouve ici des bonnes références pour les travaux qui concernent la planification du processus de désassemblage, l'achèvement de la gamme optimale, la représentation des séquences de désassemblage.

Geiger et Zussman [Geiger, 1999] ont proposé l'intégration d'un modèle dynamique à l'aide des réseaux Bayesiens, pour mettre en place une planification réactive du processus de désassemblage. Ils utilisent comme outil de modélisation du désassemblage le graphe de récupération qui est un graphe ET/OU avec des valeurs de fin de vie pour chaque sommet qui représente un composant ou un sous-ensemble. Une probabilité de succès est associée à la chaque opération de désassemblage. La méthode de la planification réactive est testée sur un poste de radio obsolète.

Dans [Imtanavanich, 2004] Imtanavanich et Gupta proposent une solution pour déterminer le nombre optimal de produits à désassembler pour satisfaire une certaine quantité demandée de sous-ensembles. Le problème stochastique est transformé dans un problème equivalent déterministe. La méthode de programmation utilisée est "goal programming". L'approche stochastique rend le modèle plus réaliste. Le processus de désassemblage a lui même un caractère stochastique : la condition des sous-ensembles, le pourcentage de réutilisation, le pourcentage de recyclage et le rapport entre les opérations destructives et non destructives.

La plupart des travaux est concentrée sur le domaine de la planification du processus de désassemblage. Cette fois il s'agit d'obtenir la gamme optimale de désassemblage d'habitude après un critère économique. L'article de Lambert donne des bonnes références dans ce domaine. L'auteur a examiné plus de 135 articles liés à la planification du processus de désassemblage parmi lesquels plus de 25 sont des recherches des années 2000-2002 qui concernent des méthodes pour obtenir la meilleure gamme de désassemblage. La plupart des méthodes de planification de processus de désassemblage sont graphiques, heuristiques, tiennent de domaine de l'analyse multicritère ou de la programmation mathématique et de l'intelligence artificielle.

### 2.4.3. Etat de l'art de l'ordonnancement du désassemblage

Un essai pour traiter le problème de l'ordonnancement flexible a été fait dans l'article de Wiendahl [Wiendhal, 2001] qui a proposé une architecture de type anneau pour la ligne de désassemblage. Les auteurs ont illustré leur approche par la réalisation d'un système modulaire pour le désassemblage de trois produits. Une cellule de désassemblage a été réalisée à l'Institut Technion de Haifa. La cellule est contrôlée par un module d'analyse de données. Les temps opératoires sont non déterministes et dans une distribution aléatoire. Le processus n'est pas stable. L'échec du désassemblage conduit à reprendre les opérations.

Dans [Gupta, 1994] Gupta et Taleb ont proposé un nouvel algorithme pour l'ordonnancement du désassemblage pour des produits avec une structure bien déterminée. L'algorithme présenté donne un plan de désassemblage pour les composants du produit en fonction de la demande. L'algorithme est de type MDS (*Multiple Demande Sources*). L'approche de Gupta ne prend pas en considération les aléas de désassemblage. Ce processus est supposé être parfait c'est à dire que le désassemblage est non destructif.

L'utilisation d'un système flexible Kanban a été proposée par Kizilkaya et Gupta dans [Kizilkaya, 1998]. Pour contrôler le flux de matériaux dans la cellule de désassemblage la méthode proposée trouve le pourcentage de la demande qui stabilise le flux dans le système. Les résultats ont été obtenus par une simulation à l'aide du logiciel SIMAN. Les hypothèses de l'ordonnancement sont restrictives: le produit est complètement désassemblé, la ligne de désassemblage est en série, et la demande est constante pendant le processus. Taleb et Gupta ont proposé un algorithme pour établir l'ordre optimal des opérations en fonction de la demande dans le désassemblage des produits avec des matériaux communs [Taleb, 1998].

Dans les dernières années les recherches de l'équipe de professeur Gupta se sont concentrées sur l'aspect *d'équilibrage des lignes de désassemblage*.

Même si le problème d'équilibrage des lignes est connu depuis longtemps, aucune étude n'a pas été faite sur l'équilibrage des lignes de désassemblage jusque à [Gungor, 1999a] [Gungor, 1999b]. Gungor et Gupta ont traité le cas du problème d'équilibrage simple d'une ligne de désassemblage (*Simple Disassembly Line Balancing Problem*) (DLBP-S) [Gungor, 1999b]. Ils on supposé qu'il y a un stock infini d'entrée et que la configuration du produit n'est pas changée à la fin de sa vie. Pour simplifier le problème de DLBP les auteurs ont utilisé dans leur algorithme des temps opératoires déterministes et connus. La demande de composants est aussi connue et l'horizon de planification est un jour. L'objectif du DLBP-S est une utilisation efficace des ressources de la ligne de désassemblage qui satisfait la demande. Dans le même temps les relations de précédences entre les tâches doivent être satisfaites pendant leur affectation. Dans le cas de l'assemblage, les relations géométriques ainsi que les relations fonctionnelles entre les composants sont importantes. Par contre, dans le désassemblage la fonctionnalité des composants séparés n'est pas prise en compte. Il reste les liens géométriques et de contact entre les sous-ensembles du produit. Gungor et Gupta ont illustré leur

algorithme par le désassemblage d'un ordinateur fixe. Il s'agit d'un algorithme heuristique qui donne l'affectation optimale des tâches de désassemblage sur huit postes de travail [Gungor, 1999b]. Le critère d'optimisation pour réaliser l'équilibrage de la ligne a été de minimiser le temps d'inactivité de chaque station. Dans ce travail la définition du problème de l'équilibrage des lignes de désassemblage ressemble à celle du problème ALBP. Mais il y a plusieurs différences importantes entre les deux processus comme on l'a vu au paragraphe 1.4., alors nous considérons leur approche incomplète.

L'algorithme utilisé en [Gungor, 1999b] a une complexité réduite. De plus, la méthode ne dépend pas du nombre de stations de désassemblage ou de la demande. La méthode a encore un point faible : les auteurs n'ont pas pris en considération les incertitudes de désassemblage.

Dans [Gungor, 2001] les mêmes auteurs proposent une solution pour le problème d'équilibrage de la ligne prenant en considération les échecs des opérations de désassemblage. Il s'agit d'affecter les tâches aux postes de sorte que l'effet des échecs opératoires sur la ligne de désassemblage soit minimisé. Les auteurs ont fait une comparaison entre les processus d'assemblage et désassemblage pour ce qui concerne la variabilité de la demande, l'incertitude pendant le processus, la complexité des opérations et les méthodes d'optimisation du fonctionnement de la ligne. Le désassemblage est soumis à plusieurs contraintes de précédence entre les composants: les relations prises en considération sont de type AND, OR, et de type AND/OR. Les situations qui apparaissent après l'échec de désassemblage sont: le produit part plus tôt ou plus tard du poste de travail, le produit est détourné d'un poste, le produit visite un poste plusieurs fois. La méthode proposée a plusieurs étapes:

- générer le séquencement complet des opérations qui prend en considération l'échec éventuel de certaines tâches;
- choisir les séquences qui minimisent les temps morts;
- associer à chaque tâche un coût d'échec avec une certaine probabilité;
- trouver l'affectation des tâches qui minimise les coûts (à l'aide de l'algorithme de Dijkstra).

Néanmoins, la méthode proposée par Gupta donne un algorithme optimal par rapport aux coûts opératoires, mais elle ne prend pas en considération les valeurs en fin de vie des composants récupérés. Dans les processus réels de désassemblage on préfère souvent un désassemblage destructif au lieu d'un désassemblage propre pour minimiser le coût du processus.

Dans [McGovern, 2004] McGovern et Gupta proposent deux méthodes d'optimisation combinatoire pour l'équilibrage de la ligne de désassemblage : les algorithmes génétiques et les méta heuristiques Hunter Killer. Les auteurs utilisent une technique de recherche exhaustive dans l'espace des solutions pour trouver une solution optimale et pour établir des temps de travail égaux. Le but de la méthode est de minimiser le nombre de stations de travail donc leur technique peut être utilisée dans l'étape de la conception de la ligne de désassemblage. La méthode est approprié au problème DLBP déterministe. Les algorithmes génétiques sont rèlativement lents. La meta heuristique trouve des sous-solutions optimales convenables. L'avantage de la méthode proposée dans cet article est que pour un problème de taille reduite, c'est à dire d'un produit qui

nécessite un nombre d'opérations de désassemblage n<12, la solution trouvée est trés proche de la solution optimale. Une combinaison des methodes proposées est envisagé pour augumenter la taille du problème.

### 2.4.4. Approche commande du processus de désassemblage

Dans sa thèse, Chevron [Chevron, 1999] propose un outil de modélisation qui peut représenter le comportement d'une cellule de désassemblage basé sur les réseaux de Petri. Son but est de superviser et de contrôler la cellule. L'auteur est parti du modèle connu du produit. Il a utilisé comme outil de modélisation les réseaux de Petri Colorés Interprétés, qui lui ont permis de modéliser différentes fonctions de la cellule au niveau décisionnel et opérationnel. Il a utilisé les RdPCI pour modéliser les contraintes techniques de démontage. Les couleurs des places représentent des attributs comme les temps opératoires ou des données quantitatives ou qualitatives.

En ce qui concerne le contrôle, Chevron a utilisé un ordinateur qui lui a permis de contrôler en ligne le procédé de démontage et un robot dont la gestuelle correspond aux différentes opérations. Dans une première phase il a simulé la cellule de démontage par deux ordinateurs et un robot. Un ordinateur permet la programmation hors ligne des macro opérations sur le contrôleur du robot et l'autre ordinateur envoie une commande en ligne qui correspond à la progression du démontage. La cellule a été réalisée au laboratoire d'automatique de Grenoble.

Dans son travail D. Chevron n'a pas pris en considération la possibilité d'échec dans une opération de désassemblage. Si le produit à désassembler est un produit à la fin de sa vie alors il est très probable que ses composants ne sont plus dans un bon état. On peut estimer que certaines opérations de désassemblage ne seront plus réalisables donc une analyse prévisionnelle est nécessaire. En plus, les temps opérationnels de désassemblage dans le cas des composants détériorés ne sont pas les mêmes que les temps prévus dans l'étape de conception du produit pour le désassemblage. L'une des difficultés de représentation de la cellule de désassemblage est l'incertitude relative à l'état du produit. On doit utiliser alors un outil de modélisation qui peut prendre en considération l'incertitude et l'échec de certaines opérations de désassemblage.

S. Puente dans [Puente, 2002] a réalisé une cellule de désassemblage pour le démontage non destructif des ordinateurs de type PC. Le produit est fixé sur une table rotative. La cellule est formée par un robot avec 5 degrés de liberté qui exécute les opérations de démontage dictées par un système informatique. Ce système est formé par un système de vision artificielle qui permet la reconnaissance du produit et de ses composants, un planificateur qui donne les séquences de désassemblage, un programme de commande du robot avec une interface réalisée en Visual C++. La fusion de données du système de vision artificielle avec la base de données initiale donne au planificateur un modèle particulier du produit qui sera soumis au désassemblage. Ce modèle géométrique est utilisé pour calculer la trajectoire du composant qui sera séparé du reste du produit. Après, une simulation est effectuée, dans le but de déterminer les collisions éventuelles. Si c'est le cas, la configuration de la commande est changée pour permettre d'enlever un autre composant.

Le problème apparaît quand un ou plusieurs composants ne peuvent pas être démontés sans destruction. L'auteur a précisé dans son travail que son approche est efficace pour des produits qui ne sont pas trop détériorés à la fin de leur vie. Ainsi, il utilise la cellule de désassemblage pour le démontage des ordinateurs de type ordinateurs portables PC, ordinateurs qui ne subissent pratiquement pas de modifications physiques pendant leur utilisation. L'avantage du système du Puente est évidement la fusion d'informations données par le système de vision en temps réel avec le modèle géométrique établi dans la phase de conception du produit. La cellule conçue par Puente permet la réalisation des tâches parallèles ou l'utilisation des robots coopératifs dans certaines tâches de désassemblage [Puente, 2003].

R. Fernandez dans [Fernandez, 2001] a proposé un modèle pour commander un atelier de désassemblage dans lequel toutes les opérations sont effectuées par des stations différentes. Il a utilisé comme outil de modélisation les Réseaux de Petri Continus qui lui ont permis de prendre en considération l'aspect temporel du problème de désassemblage. Le modèle de Fernandez permet d'établir une loi de commande à partir de la donnée d'un ensemble d'arbres de désassemblage représenté par un réseau Petri continu. La méthode permet de déterminer l'évolution des marquages dans le temps donc le comportement asymptotique du système de désassemblage. Ce modèle est convenable quand les opérations qui n'ont pas de relation de précédence sont effectuées en parallèle.

Kizilkaya [Kizilkaya, 1998] a proposé un système Kanban flexible pour contrôler le flux de matériel dans une cellule de désassemblage. Le système Kanban flexible permet de faire varier le nombre de Kanbans au cours d'un cycle de désassemblage pour compenser les erreurs dues aux incertitudes. La cellule de désassemblage est composée de N stations en ligne qui ont chacune un opérateur humain. Le flux matériel est contrôlé par le nombre de Kanbans de production. Lorsque la demande est satisfaite, le module du système Kanban flexible commence par enlever des Kanbans du système. Pour déterminer par simulation la quantité convenable pour la demande optimale d'un jour, les auteurs ont utilisé l'application SIMAN V. Le système flexible a comme avantage la réalisation de la demande d'un composant particulier même s'il faut désassembler plusieurs unités du produit pour satisfaire cette demande. En tenant compte que la méthode Kanban a été conçue pour les environnements stables, le système proposé ne pas applicable si certains opérations ne sont pas faisables. En fait, une des hypothèses est que le produit est complètement désassemblé.

Dans son article Kopacek [Kopacek, 2003] a présenté une cellule de désassemblage pour les téléphones portables. La cellule est semi-automatique et est formée de six stations en ligne: une station manuelle pour le chargement et le déchargement de la cellule, trois stations de perçage et deux pour enlever les carcasses et le circuit électronique. Les robots utilisés sont cartésiens. Un système hiérarchique de contrôleurs programmés par un ordinateur forme le module de contrôle. La commande est réalisée par cet ordinateur central. La tâche principale de l'ordinateur de contrôle est de transférer le modèle géométrique du produit vers les stations et assurer la gestion des

signaux des capteurs. Le logiciel pour la commande est écrit en C++ avec une interface simple.

La cellule a été conçue seulement pour récupérer la carte électronique de base d'un seul modèle de téléphone portable. Si le modèle change, l'opérateur humain doit changer les outils qui sont actionnés par les robots pour le désassemblage. De plus, le désassemblage est non destructif parce que les portables n'ont pas plus de trois ans et ils n'ont pas souffert de beaucoup de dégradations pendant leur fonctionnement. Une autre difficulté qui intervient dans le désassemblage des téléphones portables c'est que les fabricants utilisent différents matériels et une grande variété de formes, ce qui demande un système de désassemblage très flexible.

Wiendahl et Selinger [Wiendhal, 1999] ont présenté dans leur travail une approche particulière pour concevoir un système de planification et de contrôle pour le processus de désassemblage. Dans leur étude, le système de contrôle et commande doit contenir des modules pour l'acquisition et l'interprétation d'informations sur le produit, ainsi qu'un module pour le contrôle flou dans le système de désassemblage. Du fait des particularités du désassemblage pour chaque produit et/ou composant, un module d'aide à la décision doit être inclus dans le système de contrôle. Les sorties de ce module sont utilisées pour établir l'affectation des tâches sur les machines. Les auteurs ont suggéré la programmation logique pour programmer ce type de module de commande.

Un système d'aide à la décision a été aussi proposé par Chevron dans sa thèse [Chevron, 1999]. Il propose un SIAD qui a plusieurs fonctions: générer les séquences de démontage, évaluer les performances de ces séquences et les risques d'échecs associés, sélectionner parmi ces séquences l'ensemble qui répond le mieux au critères considérés (revenu maximal, temps opératoire minimal). Le modèle analysé par le système d'aide à la décision a été représenté par des réseaux de Petri Colorés interprètés. Dans ce cas, les RPCI de contrôle présentent l'intérêt de caractériser le comportement de la cellule de démontage. Une autre caractéristique du travail de Chevron est l'adaptation de ces réseaux de telle façon qu'ils intègrent la dimension aléatoire des temps opératoires de démontage.

Pham V. H. a proposé dans son travail [Pham, 1999] une méthode pour la conduite des processus de désassemblage basée sur une approche dynamique permettant la réaction en temps réel aux événements imprévus. Il a défini un système de pilotage en temps réel architecturé multi agents pour une bonne tolérance aux aléas du désassemblage. Le système multi agents permet de simplifier la prise de décision pour la sélection dynamique de la gamme par le simple jeu d'une concurrence entre agents.

## 2.5. Conclusions

Dans ce chapitre nous avons montré que la commande des systèmes d'assemblage est hiérarchisée en plusieurs niveaux:

- *Planification:* définition de l'ensemble des produits à assembler par type de produits sur un horizon H, divisé en périodes de durée T (généralement la semaine)
- *Ordonnancement hors ligne:* définition, pour chaque période T, de l'ordre de passage des produits et affectations des tâches aux postes
- *Ordonnancement en ligne:* affectation en temps réel des tâches; dans certains cas on peut aussi intervenir sur l'ordre de passage des produits, mais cette possibilité d'action en temps réel est limitée ;
- *Pilotage:* envoi des ordres aux équipements

Nous ne nous intéressons dans notre travail qu'aux niveaux ordonnancement. Dans le cas de l'assemblage et du désassemblage, la problématique rejoint en partie celle de la conception, les ressources du système et la charge de travail sur un horizon T (pour simplifier nous dirons sur la semaine) étant maintenant déterminées. L'objectif ici est l'optimisation de l'emploi des ressources, ce qui se traduit généralement par la minimisation du temps de travail ou encore, en manuel, par l'équilibrage du système. Pour l'ordonnancement hors ligne, les moyens d'action sont le séquencement des produits et l'affectation des tâches aux postes. Dans le cas de la *commande des systèmes de désassemblage*, nous ne considérons que le niveau de l'ordonnancement.

Pour l'ordonnancement en ligne, il est pratiquement très difficile, pour des problèmes d'encombrement de modifier le séquencement. Le seul moyen d'action est donc l'affectation, d'ailleurs très limitée comme nous l'avons rappelé ci-dessus. La très grande variabilité des temps opératoires, et plus encore la variabilité opératoire (démontage propre ou destructif) qui se découvre pour une bonne part en temps réel rend la qualité de cet ordonnancement extrêmement incertaine et nécessite une grande réactivité du système, ce qui donne une importance particulière à l'ordonnancement en temps réel. Outre l'affectation et le séquencement, les moyens d'action pour la commande sont ici : *le niveau de désassemblage* et *la méthode de démontage* (propre pour récupérer les composants ou destructive pour récupérer la matière). Ces deux éléments sont corrélés: on désassemble d'autant plus qu'on cherche à récupérer davantage de pièces en bon état ce qui requiert un démontage soigné.

Ainsi, nous allons étudier dans le chapitre suivant les deux niveaux de la commande: **l'ordonnancement hors ligne** et **l'ordonnancement en ligne.** Le problème d'équilibrage sera traité en conjonction avec l'ordonnancement. Dans notre travail nous ne considerons que le problème d'équilibrage d'une ligne avec un nombre fixé de stations. Notre travail ne concerne pas la phase de conception de la ligne. Le problème devient plus difficile s'il s'agit d'équilibrage des lignes de désassemblage. Les particularités de ce problème seront présentées dans le chapitre d'optimisation. De plus, le problème d'équilibrage avec $n$ fixé peut avoir deux objectifs: minimiser le temps de cycle ou équilibrer les charges sur les postes. Puisque ces objectifs sont différents, les résultats sont en général différents. Le premier cas peut donner comme résultat une ligne non équilibrée pendant que le deuxième donne un bon équilibrage.

# CHAPITRE III

# L'ORDONNANCEMENT DES LIGNES DE DÉSASSEMBLAGE

## 3.1. Présentation de la problématique

La conception et la planification d'un système de désassemblage sont des étapes nécessaires pour établir la structure, les ressources, la capacité de ce système et le déroulement du processus de désassemblage. Dans ce cadre, l'ordonnancement doit assurer le flux des produits sur la ligne. L'étape d'ordonnancement est très importante dans le contrôle des temps opérationnels et en général des performances économiques du système.

La plupart des systèmes de production sont étudiés dans une approche déterministe. Cette hypothèse permet de trouver des solutions prescriptives pour le problème d'ordonnancement qui peuvent guider les décisions d'affectation des tâches. Pourtant, les systèmes de désassemblage travaillent dans un environnement fortement stochastique. Une solution pour l'ordonnancement des ces systèmes est d'appliquer périodiquement l'ordonnancement prévisionnel déterministe pour réduire les effets des aléas caractéristiques sur le processus de désassemblage. Il est préférable que l'étape de l'établissement de l'ordre des opérations permette de déterminer un ordre qui entraîne des résultats satisfaisants dans des conditions réalistes, plutôt qu'un ordre qui donne des résultats théoriques optimaux, mais selon des hypothèses trop éloignées de la réalité.

### 3.1.1. Les objectifs de l'ordonnancement

Dans un problème d'ordonnancement il est important de définir un certain nombre d'objectifs à atteindre. Il s'agit d'optimiser (maximiser ou minimiser) une fonction d'évaluation en respectant un certain nombre de contraintes. D'habitude l'évaluation de l'ordonnancement porte sur trois facettes : le délai, le coût et la qualité. Les trois paramètres ont été déjà analysés dans le paragraphe 2.2.1.4.

Néanmoins, il est rare d'associer la qualité du processus à l'évaluation de son ordonnancement. La fluidité du flux de production et le *makespan* [2] sont souvent utilisées pour l'évaluation de la qualité d'ordonnancement. Dans notre travail nous allons évaluer les temps opératoires. L'ordonnancement est donc évalué du point de vue temporel dans le but de minimiser la période de stationnement du produit sur un poste de désassemblage. Dans le désassemblage la technique d'ordonnancement utilisée doit prendre en compte le fait que chaque composant ou sous ensemble est une source de demande et en plus il peut être détérioré après l'usage du produit. Ainsi l'ordonnancement doit prendre en considération les opérations et les ressources nécessaires pour obtenir le composant désiré même dans le cas des opérations destructives de désassemblage. Pendant le désassemblage beaucoup des problèmes peuvent apparaître à cause de l'incertitude sur l'état de chaque composant du produit traité. Ce fait nécessite la réalisation d'un système de contrôle flexible qui aurait éventuellement la capacité de réaliser un ré ordonnancement en ligne des tâches de désassemblage. C'est le problème le plus difficile dans la problématique de l'ordonnancement du désassemblage pour lequel on proposera des solutions dans les paragraphes suivants.

### 3.1.2. Les catégories des problèmes d'ordonnancement

Les problèmes d'ordonnancement peuvent être classés en deux catégories en fonction du nombre de machines nécessaires pour réaliser chaque tâche. La première catégorie regroupe les problèmes pour lesquels chaque tâche nécessite une seule machine et la deuxième ceux pour lesquels chaque tâche demande plusieurs machines pour son exécution. La première catégorie concerne les ateliers dans lesquels plusieurs machines sont disponibles pour l'exécution d'un travail ou les ateliers à machines parallèles. La deuxième classe de problèmes concerne les ateliers à *m* stations de travail différentes. Une station comporte une ou plusieurs machines en parallèle. Les tâches à réaliser sont constituées d'un ensemble de *n* opérations, chacune d'elles peut s'exécuter sur la même machine ou sur une machine différente de celles d'autres opérations. En fonction de l'ordre de passage des opérations sur les machines (appelé aussi gamme opératoire) il y a trois typologies d'atelier : *le flow shop*, *le job shop* et *le open shop*.

#### 3.1.2.1. Flow shop

Le flow shop se rencontre dans les ateliers disposant de lignes de production dédiées à la production de masse de peu de variétés de produits. Un tel atelier est aussi appelé "en ligne" ou à cheminement unique. Chaque tâche est constituée de n opérations et chacune de ces opérations doit être exécutée sur une machine différente. L'ordre de passage des opérations sur les machines est le même pour toutes les tâches. Il y a plusieurs types de flow shops qui diffèrent par le type des gammes opératoires. On parle de flow shop hybride s'il est possible de trouver plusieurs machines pour réaliser une opération donnée. Le flow shop de permutation consiste à affecter les tâches dans un ordre identique pour toutes les machines.

---

[2] Le makespan est la date de fin de la dernière opération exécutée.

3.1.2.2. Job shop

Dans ce type d'atelier, chaque tâche possède son propre mode de passage sur les machines. Suivant les gammes opératoires le cheminement des opérations ne sera pas le même.

3.1.2.3. Open shop

Dans l'atelier open shop le cheminement des opérations est multiple, mais à la différence du job shop les jobs ne possèdent pas des gammes. L'ordre de passage des opérations est quelconque. Le cheminement des opérations est ainsi déterminé par l'ordonnancement.

Dans notre travail le problème d'ordonnancement est de type *flow-shop*. Ce type de problème est un cas particulier du problème de base d'ordonnancement d'atelier pour lequel le cheminement est unique : les $n$ produits sont traités sur les $m$ machines (ou entrent dans les $m$ postes de travail) dans l'ordre $1,\ldots, n$. Dans le cas de l'atelier à cheminement unique tout travail visite chaque machine de l'atelier et l'ordre de passage d'un travail sur les différentes machines est le même pour touts les travaux (flot unidirectionnel). Cet ordre unique est une donnée du problème. Dans le cas du flow-shop simplifié ou de permutation la séquence des travaux visitant une machine est la même pour toutes les machines. Le cas du flow-shop hybride correspond à une généralisation des problèmes de flow-shop et de machines en parallèle. L'atelier est constitué d'un certain nombre d'étages en série, chaque étage étant composé de plusieurs machines en parallèle. Les résultats présentés ici se limitent au flow-shop de permutation.

En effet, l'approche flow-shop simplifié est la plus approprié pour étudier le comportement des lignes de désassemblage [Gungor, 2001]. Le désassemblage des produits est effectué soit sur une seule station de travail, soit dans une cellule de désassemblage, soit sur une ligne de désassemblage. L'architecture uni poste assure une grande flexibilité du poste de travail. Le désavantage principal consiste dans la nécessite d'un grande espace de stockage autour du poste pour les outils, pour les produits qui arrivent et pour les composants désassemblés. Dans le même temps la productivité d'un tel poste est relativement basse avec des temps opératoires élevés.

Dans le cas d'une cellule flexible de désassemblage, les coûts opératoires et les coûts des équipements sont grands par rapport au profit obtenu. Les investissements sont plus justifiés dans le cas d'une ligne de désassemblage qui assure une grande productivité. En plus, l'architecture flow-shop est plus appropriée au désassemblage de grands produits (les voitures) ou au désassemblage de grandes séries pour les petits produits (les téléphones portables). Pourtant, l'architecture en série de la ligne de désassemblage est convenable pour le désassemblage automatisé. Dans le cas général de l'ordonnancement flow-shop il y a de tâches qui peuvent être déplacées d'un poste voisin à l'autre respectant toutefois les contraintes de précédences. Dans le cas du désassemblage

automatisé le flux des opérations est linéaire sans retours aux postes déjà visités par le produit.

Ce sont ces critères qui ont déterminé de choisir une architecture flow-shop de la ligne de désassemblage pour laquelle nous étudierons deux cas : le désassemblage manuel et le désassemblage automatisé.

Dans le processus d'assemblage l'ordonnancement des tâches est fait en vue d'assurer un bon équilibrage de la ligne et de baisser le temps entre l'entrée des composants et la sortie du produit fini. Bien que des nombreux chercheurs aient traité le problème d'ordonnancement et d'équilibrage des lignes d'assemblage depuis une quarantaine d'années [Scholl, 2003], il y a encore des aspects pratiques qui restent sans solution. La formulation du problème d'ordonnancement est basée sur l'hypothèse que le processus est déterministe. Cette hypothèse permet la résolution par une variété des méthodes prescriptives qui donnent le cadre général pour la prise des décisions dans l'ordonnancement. Pourtant, les systèmes réels travaillent dans un environnement dynamique, parfois stochastique. Une solution est d'appliquer le modèle déterministe périodiquement pour compenser les événements aléatoires comme l'interruption du fonctionnement d'une machine ou l'arrivé des nouvelles tâches. Une autre solution est d'utiliser des algorithmes stochastiques qui fournissent une bonne solution pour un moment donné.

### 3.1.3. L'ordonnancement hors ligne (prévisionnel)

Dans le Chapitre 2 nous avons fait une présentation de la notion d'ordonnancement prévisionnel. Elle consiste en trois étapes : la recherche des gammes qui tient compte de toutes les contraintes générées par le produit, l'affectation des tâches aux postes de travail qui tient compte des ressources disponibles pour accomplir les tâches et le sequencement des produits.

#### 3.1.3.1. La plannification des opérations

Dans la plupart des systèmes de production l'ordonnancement est une étape facile dès que les gammes opératoires sont établis. Une séquence pour un ensemble d'opérations décrit l'ordre dans laquelle les opérations sont accomplies dans le système. La planification des opérations est fait *avant* l'ordonnancement puisque ce dernier donne les dates de la réalisation des opérations ordonnées. C'est le rôle du planner d'établir les gammes de désassemblage. Dans les systèmes de désassemblage conçus pour un œul type de produit, la planification des gammes doit prendre en compte les contraintes physiques et fonctionnelles entre les composants de ce produit. Plusieurs travaux ont traité ce problème.

Dans les systèmes de désassemblage pour les familles de produits il s'agit du problème du *séquencement des types de produits* de la ligne de production qui consiste à déterminer l'ordre d'entrée des produits sur la ligne. Cette problématique est très bien

connue pour les lignes multi produits, comme par exemple dans le cas de l'ordonnancement d'une ligne d'assemblage de véhicules multi modèles [Lopez, 2001]. La question est dans quel ordre faut-il lancer les véhicules sur la ligne de façon à optimiser le processus de production et à satisfaire les commandes fermes des clients? Dans ce cas le problème de l'ordonnancement se restreint au choix des véhicules à lancer en entrée de ligne à des instants prédéfinis et donc au choix de la séquence. Ainsi, pour les familles de produits le problème est à quel instant est lancé sur la ligne un certain type de produit qui appartient à la famille. C'est pourquoi on parle dans certains cas de "*cadencement*".

Comme dans notre travail il y a des exemples qui concernent le désassemblage de voitures, nous sommes intéressés de trouver la meilleure séquence pour une famille de produits. Les différents modèles n'ont pas la même charge de travail et donc l'ordonnancement a pour but la régularisation de cette charge pour tous les postes. Pour chaque voiture il y a des tâches constantes et des tâches variables. Les tâches constantes sont présentes pour toute la famille de voitures et ont le même temps opératoire (par exemple de démontage du pare-brise). La variabilité des tâches dérive soit du temps de travail différent nécessaire pour le désassemblage de certains sous-ensembles selon une méthode destructive ou non, soit de la présence du sous-ensemble dans le type de voiture (la climatisation, le lève-vitre automatique, les airbags, le toit ouvrant). Ces tâches variables induisent une charge variable sur les postes. Si le séquencement n'est pas bien fait les opérateurs prendront des retards dans leur travail et on aura des périodes d'inactivité [Lopez, 2001]. Ici le problème de séquencement consiste à déterminer dans quel ordre il faut lancer les véhicules sur la ligne de façon à optimiser le processus de désassemblage. Cette approche prévisionnelle présente deux avantages : elle permet de pousser le flux et de définir à l'avance les commandes fermes pour les fournisseurs de véhicules hors usage.

Mais suite aux aléas du processus de désassemblage ce séquencement prévisionnel ne peut jamais être appliqué intégralement. Les tâches destructives et les tâches variables selon le modèle de produit provoquent des perturbations du séquencement prévu. Il est alors nécessaire d'envisager un ordonnancement dynamique.

Pour faire face à ce problème nous allons utiliser une méthode basée sur la détermination d'un voisinage de la charge qui donne des bons résultats mais qui conduit à un résultat sous optimal pour les problèmes de grande dimension. (A titre indicatif un problème de grande dimension est, par exemple, l'ordonnancement d'un groupe de plus de 100 voitures avec au moins 10 options).

#### 3.1.3.2. L'affectation des opérations

Le but de cette étape est de distribuer les opérations aux postes de travail pour que le désassemblage soit réalisé d'une manière optimale par rapport à un certain critère. Dans une première étape nous sommes intéressés à minimiser les coûts de désassemblage. Dans la littérature qui concerne les lignes d'assemblage le critère de coût s'exprime en

termes d'équilibrage de la ligne. Un bon équilibrage assure une charge égale sur l'ensemble de postes, des temps morts nuls donc une efficacité maximale de la ligne.

Ce point de vue est également valable dans le cas des lignes de désassemblage. Le problème d'équilibrage de la ligne de désassemblage peut être énoncé sous la forme suivante [Gungor, 1999b].

**Définition 3**
*L'équilibrage d'une ligne de désassemblage consiste à affecter n tâches de désassemblage d'un produit aux m postes de travail de façon optimale du point de vue du temps de travail et sans violer les relations de précédences entre les tâches.*

A ce point l'équilibrage est prévisionnel et le désassemblage est non destructif. Dans les systèmes réels, pendant le processus de désassemblage, il y a des opérations qui échouent à cause des déformations ou manque des composants du produit usé. Dans cette situation l'équilibrage des lignes de désassemblage peut être défini comme dans [Duta, 2005] :

**Définition 4**
*Le problème DLBP consiste dans l'affectation d'un certain nombre de tâches de désassemblage avec des temps opératoires individuels à un certain nombre de postes de travail, en tenant compte de trois aspects: de l'équilibrage des charges, des contraintes de précédences et du type d'opération de désassemblage : destructive ou non.*

Sur les lignes de désassemblage le but de l'équilibrage est d'égaliser les charges sur les postes même quand des opérations destructives sont rendues nécessaires. Dans ce cas, établir l'équilibrage devient un processus de prise de décision: quelle sorte d'opération est faite sur quel poste de façon à minimiser le temps total de désassemblage? Minimiser le temps est équivalent à minimiser le coût. Pour ce motif, l'équilibrage des lignes de désassemblage est devenu un sujet fréquent de recherche dans les dernières années.

### 3.1.4. L'ordonnancement en temps réel (réactif)

L'incertitude est le plus grand problème dans le processus de désassemblage. Les états des composants du produit arrivant dans un système de désassemblage sont parfois inconnus. Il y a un grand degré d'incertitude dans la structure et la qualité des produits usés. Certains composants ou sous-ensembles sont en bon état pendant que d'autres ont un niveau de dégradation qui les rend non démontables ou non intéressants à démonter parce que ne pouvant être réutilisés que sous forme de la matière ou d'énergie. La possibilité qu'un composant n'existe plus dans le produit doit être aussi considérée. Ces situations peuvent être prévues si la ligne est dédiée au désassemblage d'un certain type de produits.

Une autre complication peut survenir au niveau de la demande, certains composants pouvant être ou ne pas être demandés. Ces considérations peuvent affecter autant le nombre de produits à désassembler que l'équilibrage de la ligne. Dans ces conditions la caractéristique principale du système de désassemblage est sa grande flexibilité qui lui permet de s'adapter aux incertitudes sur l'état du produit traité aussi bien qu'au fonctionnement des ressources du système [Addouche, 2003].

L'ordre des opérations est établi à l'étape précédente. Ainsi, on ne pourra pas modifier l'ordre des opérations, mais on pourra modifier leurs affectations en fonction de l'avancement du processus de désassemblage.

L'ordonnancement en temps réel peut être mis en oeuvre par un système interactif d'aide à la décision (SIAD). Le SIAD reçoit des informations en temps réel (les gammes de désassemblage, l'état d'un composant, la position d'un sous ensemble), il analyse les données et si l'ordonnancement prévisionnel établi avant ne peut pas être respecté, il donne des solutions pour un ré ordonnancement en fonction de la situation courante. Cette méthode a été déjà utilisée pour l'ordonnancement des lignes d'assemblage [Lopez, 2001]. Dés qu'un ensemble d'ordonnancements définis hors ligne devient non admissible, le système teste l'effet d'une ou plusieurs décisions sur le regroupement des tâches dans les machines/postes de travail, et donne éventuellement une solution optimale. C'est le rôle du décideur (l'homme ou le système de commande) de choisir la meilleur solution pour que le processus ne soit pas interrompu. Une proposition d'architecture de SIAD pour l'ordonnancement des lignes de désassemblage sera faite à la fin de ce travail.

Le problème majeur dans un processus de désassemblage est l'apparition d'un événement imprévu dans le processus de séparation des composants. Il y a des opérations qui ne peuvent pas être accomplies dans un mode propre dû aux dégradations physiques des composants. Alors il faut les faire dans un mode destructif mais parfois de telles opérations ne sont pas profitables. Dans ce contexte on doit décider la profondeur du processus de désassemblage en vue de maximiser le revenu obtenu par la valorisation des composants.

Les fabricants de l'industrie du démontage veulent accomplir trois objectifs majeurs: des coûts inférieurs, des plus grands revenus et des lignes équilibrées. On peut donc regarder le problème d'evaluation du processus de désassemblage sur deux aspects: l'équilibrage de la ligne et le coût du désassemblage. Dans les paragraphes qui suivent chaque aspect sera détailé avec ses problèmes qui en rendent l'optimisation difficile.

## 3.2. Evaluation de l'ordonnancement du désassemblage par rapport à l'équilibrage de la ligne

### 3.2.1. L'équilibrage des lignes d'assemblage

#### 3.2.1.1. La problématique

L'équilibrage de la ligne est une approche traditionnelle qui a pour but une affectation des tâches aux postes de travail conduisant à un équilibrage de leurs charges. Sur une ligne équilibré la productivité augmente, le temps de travail est mieux utilisé, les variations des demandes sont compensées, le coût de production baisse. Dans ce sens, l'équilibrage de la ligne peut être considéré comme une technique d'ordonnancement utilisée dans le contrôle de la production [Nof, 1997].

L'équilibrage des lignes de fabrication concerne essentiellement *les lignes d'assemblage*. Les tâches sur une ligne d'assemblage sont ordonnées. La ligne d'assemblage est composée par plusieurs postes connectés par un convoyeur ou par d'autres types de dispositifs de transfert. Les tâches exécutées sont plus ou moins complexes. Le but est d'assurer une utilisation maximale et un temps de travail presque égal sur chaque poste. Une ligne équilibrée fournit un flot de production sans délai. Le problème de l'équilibrage se pose soit au moment de la conception de la ligne, soit en cours de fonctionnement lorsqu'il y a des variations de charges importantes et qu'il faut modifier la cadence de production. Pour une ligne manuelle, le coût de la ligne est directement lié au nombre des stations de travail. Alors, dans la phase de la conception de la ligne, le but de l'équilibrage est de minimiser le nombre de postes. Sur les lignes automatisées le coût et le temps opératoire dépendent de la ressource utilisée. Dans ce cas le problème est quelle tâche est affectée à quel poste.

La ligne d'assemblage est caractérisée par le déplacement des produits d'un poste à l'autre. Les produits restent un certain temps dans un certain poste de travail. Ce temps est appelé *le temps de cycle*. Dans l'assemblage le **temps de cycle** est défini comme *la durée majeure qui sépare deux sorties successives de produits et il est égal à la durée de travail du poste le plus lent*.

Dans le cas idéal *le temps de travail* est la période entre l'entrée du produit dans la station de travail *i* et la sortie ou la somme des temps opératoires sur cette station :

$$T_i = \sum_{j=1}^{m_i} t_j \qquad (2)$$

Où $m_i$ est le nombre d'opérations affectées au poste $W_i$ et $t_j$ le temps opératoire de l'opération $O_j$. Le temps de cycle est la valeur maximale de $T_i$

$$t_{cy} = \max_i T_i \tag{3}$$

Le temps d'inactivité $I_i$ d'un poste $W_i$ est la différence entre le temps de cycle et la somme des temps opératoires sur ce poste :

$$I_i = t_{cy} - \sum_{j=1}^{m_i} t_j \tag{4}$$

La situation idéale, donc *une ligne parfaitement équilibrée*, s'obtient quand pour n'importe quel poste *i* le temps d'inactivité *est nul.*

Pour une ligne équilibrée avec *n* postes de travail sans de temps d'inactivité, l'égalité suivante doit être vérifié :

$$T_1 = T_2 = T_3 = ... = T_n = t_{cy} \tag{5}$$

Le temps total de fabrication pour la ligne entière sera :

$$T_{total} = \sum_{i=1}^{n} T_i = n \cdot t_{cy} \tag{6}$$

Alors pour le cas où la ligne est parfaitement équilibrée, on peut déterminer le nombre de postes n si $t_{cy}$ est donné par le rapport entier :

$$n = \left[ \frac{T_{total}}{t_{cy}} \right] \qquad (7)$$

3.2.1.2. Les objectifs d'équilibrage de la ligne d'assemblage

En fonction du critère considéré, pour les lignes d'assemblage, cette optimisation a un des **objectifs** suivants :

**Obj1 : La minimisation du nombre de postes.** Dans ce cas on parle du problème simple d'équilibrage, connu dans la littérature sous le nom de problème simple d'équilibrage de premier ordre : **SALBP-1** (*Simple Assembly Line Balancing Problem*). Cette méthode s'applique aux lignes mono produites, dont les temps opératoires sont connus et déterministes. Le temps de cycle $t_{cy}$ est aussi connu et constant. L'équation (8) est déduite des équations (2), (6) et (7) et exprime le but de cette approche : *n* est le plus petit entièr donné par le rapport entre le temps total de travail et le temps de cycle.

$$n = \min_{i} \left[ \frac{\sum_{i=1} \left( \sum_{j=1}^{m_i} t_j \right)}{t_{cy}} \right] \qquad (8)$$

**Obj2 : La minimisation du temps de cycle.** Dans ce cas le but de l'équilibrage est de minimiser le temps de cycle $t_{cy}$ quand le nombre de postes est connu. C'est le problème

d'équilibrage de deuxième ordre **SALBP–2.** Dans ce cas la valeur minimale du temps de cycle est calculé en fonction de l'affectation des tâches sur les postes de travail *Wi* :

$$t_{cy} = \min[\max_{i}(\sum_{j=1}^{m_i} t_j)] \qquad (9)$$

**Obj4 : Equilibrer les charges sur les postes :** c'est un problème connu sous le nom de **EPALP** (*Equal Piles for Assembly Line Problem*). Pour un *n* donné on cherche à équilibrer les charges sur les machines. Au contraire de la méthode SALBP-2, cette méthode ne cherche pas à minimiser le temps de cycle. Le problème des piles égales (EPALP) propose d'affecter les tâches à un nombre fixé de stations de telle sorte que la charge des stations soient égales [Rekiek, 2001].

Si on représente la somme de temps opératoires sur chaque station avec un rectangle, on observe qu'il y a des différences significatives entre les surfaces de ces rectangles. Tant que la ligne n'est pas équilibrée il y a aussi des différences entre les temps totaux d'activité des stations et le temps de cycle.

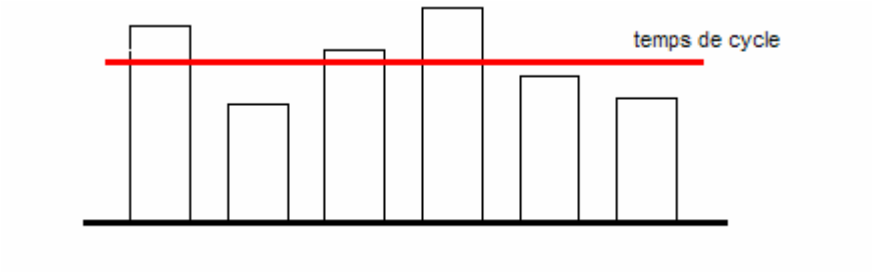



Fig. 19. Les charges sur une ligne déséquilibré

Pour minimiser ces différences il faut minimiser les surfaces qui se trouvent au-dessus et en dessous de la ligne du temps de cycle. C'est à dire qu'il faut minimiser la différence absolue :

$$\left| T_i - t_{cy} \right| \qquad (10)$$

Où $T_i$ est les temps total de travail sur un poste et son expression est donné par l'équation (2)
Pour résoudre le problème de l'équilibrage par l'approche EPAL, Rekiek travaille sur une fonction de coût à minimiser [Rekiek, 2001] :

$$f(t) = \sum_{i=1}^{n} \left( t_{cy} - T_i \right)^2 \qquad (11)$$

Où $t_{cy}$ est le temps de cycle et $T_i$ est la somme des temps opératoires sur le poste i et $n$ est le nombre de postes de travail[3].

Le problème de l'équilibrage des lignes d'assemblage ayant des groupes des opérations en parallèle a été traité dans [Dolgui, 2006]. Le but est de minimiser le coût des stations de travail. L'algorithme d'optimisation cherche le chemin le plus court dans un digraphe de précédences et il donne la solution optimale pour une ligne avec moins de cent opérations d'assemblage. Pour un nombre d'opérations qui dépasse cette valeur, l'optimisation est fait à l'aide des méthodes heuristiques. La méthode de construction du digraphe est décrite dans le cas d'une ligne en série.

### 3.2.2. L'équilibrage de la ligne de désassemblage

Dans ce paragraphe l'étude mathématique de l'ordonnancement du désassemblage est fait par rapport à l'équilibrage de la ligne.

Dans le processus de la commande l'équilibrage de la ligne est un aspect très important. En fait c'est un problème plus important qu'en assemblage parce que en désassemblage les temps opératoires présentent une forte variabilité.

Dans notre travail on s'arrête au problème d'équilibrage d'une ligne de désassemblage avec un nombre fixé de stations. Notre travail ne concerne pas la phase de conception de la ligne. Le problème devient plus difficile s'il s'agit de l'équilibrage des lignes de désassemblage. Les particularités de ce problème seront montrées dans le chapitre d'optimisation. De plus, le problème d'équilibrage avec $n$ fixé peut avoir deux objectifs : minimiser le temps de cycle ou équilibrer les charges sur les postes. Puisque ces objectifs sont différents, les résultats sont en général différents. Le premier cas peut donner comme résultat une ligne non équilibrée pendant que le deuxième donne un bon équilibrage. L'optimisation de la fonction donnée par (11) assure l'affectation optimale des tâches pour la meilleur équilibrage de la ligne.

#### 3.2.2.1. Equilibrage hors ligne

Si on introduit l'expression de $T_i$ l'équation (11) devient :

$$f(t) = \sum_{i=1}^{n} \left( t_{cy} - \sum_{j=1}^{m_i} t_j \right)^2 \qquad (12)$$

On rappelle que $t_j$ est la durée de l'opération $O_j$.

*a. Cas mono produit*

Dans les processus réels de désassemblage on rencontre des situations où les tâches de désassemblage ne peuvent pas être accomplies du fait de la dégradation des composants du produit. L'échec du désassemblage dans une manière non-destructive peut être résolu par l'introduction d'opérations destructives alternatives.

Dans le cas mono produit il faut désassembler un seul type de produit.

Nous supposerons que à l'entrée de la ligne il y a S produits de même type qui doivent être désassemblés sur l'horizon de temps H. La dimension de cet horizon est discutée dans le paragraphe 3.1.2.

Dans ce cas on définit **le temps de cycle** comme *le rapport entre l'horizon envisagé et le nombre des produits à désassembler.*

$$t_{cy} = \frac{H}{S} \qquad (13)$$

Si les produits sont tous identiques (c'est à dire qu'ils ont la même marque et la même structure), alors les temps opératoires nécessaires pour accomplir le désassemblage total sont les mêmes pour chaque produit. On doit préciser que pour le moment on n'est pas concernés par l'état du produit en fin de sa vie. Alors, la fonction $f$ a la même forme pour chaque produit et le temps de cycle est multiplié par S. On obtient une troisième forme pour la fonction considérée :

---

[3] L'expression écrite en parenthèse peut avoir une valeur *positive* si la station est considérée rapide par rapport au temps de cycle ou *négative* si la station est lente.

$$f(t) = S \cdot \sum_{i=1}^{n} \left( t_{cy} - \sum_{j=1}^{m_i} t_j \right)^2 = \sum_{i=1}^{n} \left( S \cdot t_{cy} - S \cdot \sum_{j=1}^{m_i} t_j \right)^2 \quad (14)$$

Ainsi, dans le *cas mono produit dont les produits désassemblés sont identiques* la fonction à optimiser est:

$$f(t) = \sum_{i=1}^{n} \left( H - S \cdot \sum_{j=1}^{m_i} t_j \right)^2 \quad (15)$$

*b. Cas multi produit*

Dans l'industrie de recyclage les produits qui arrivent dans le système ne sont pas vraiment identiques. Le marché des produits récupérables est très dynamique. Des produits d'une grande diversité arrivent sur la marché de recyclage: les produits électroménagers, le matériel de bureautique, les machines industrielles, les équipements médicaux, les téléphones portables, les appareils photo, les moteurs de voitures. Le secteur industriel de l'approvisionnement du marché automobile, de l'après-vente à proprement parler, de même que des milliers d'entreprises indépendantes, remanufacturent des moteurs usés ou défectueux pour les amener aux normes des équipementiers. En moyenne, une voiture ou un camion sur dix parmi ceux qui circulent sur la route, a besoin d'un moteur de remplacement pendant sa durée de vie de plus de dix ans. Le secteur des machines lourdes a été l'un des premiers secteurs à remanufacturer ses produits. Ces produits se caractérisent par une durée de vie longue et une grande valeur ajoutée récupérable [Ontiveros, 2004].

Un système de désassemblage a la capacité d'intégrer une grande variété de produits pour la valorisation, la maintenance et la réparation. Exemple : capacité d'une unité de valorisation de véhicules hors d'usage de traiter un grand nombre de modèles de véhicules.

Il a été prouvé qu'une ligne de désassemblage pour un seul type de produit ne se justifie pas de point du vue économique [Kopacek, 1998]. Par exemple, une cellule de désassemblage pour un seul type d'ordinateur ou téléphone portable ne peut pas opérer avec profit. Le nombre de pièces ou de sous-ensembles récupérés est bas et, donc, le

revenu final obtenu peut être inférieur à l'investissement. Le problème est donc de collecter et transporter une quantité suffisante de types de produits pour satisfaire la demande de sous-ensembles sur le marché secondaire.

Une manière de simplifier cette problématique trés vaste est de grouper les produits dans des familles de produits et de trouver l'optimum à récupérer dans ce groupe.
*Les familles de produits* sont des groupes de produits similaires ou différents qui demandent presque les mêmes opérations de désassemblage faites avec les mêmes outils. Des technologies différentes de désassemblage doivent être identifiées dans le groupe des produits. Puis, pour chaque type de produit on doit identifier les opérations de désassemblage qui sont faites dans un mode destructif ou non destructif et les composants ou les sous-ensembles qui seront désassemblés. Il faut définir la notion de gamme générique de désassemblage dans le but de concevoir un système de désassemblage unique pour traiter tous les produits d'une famille donnée. Pour limiter la complexité d'analyse due au nombre de variantes de produits, la famille de produits peut être découpée dans des entités fonctionnelles. Dans ce cas le système de désassemblage sera modulaire avec des cellules pour désassembler les parties similaires des produits de la même famille.

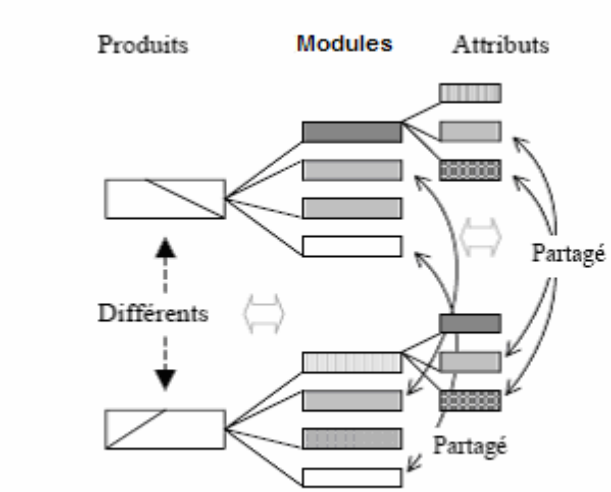



Fig. 20. Les charactéristiques modulaires des produits de la même famille

En ce point, il faut préciser les différentes opérations de désassemblage qui sont envisagées pour chaque type de produit. On prend le cas des voitures. Une famille de voitures comporte différent sous ensembles fonctionnels tels que le moteur, la carrosserie, les airbags etc. Une entité peut être constitué de variantes ainsi le“ moteur” a pour variantes : moteur à essence, moteur diesel. D’autre part, une entité peut être commune ou spécifique. Elle est commune si elle intervient dans la composition de tous les produits de la famille et elle est spécifique si elle est désirée par le client (optionnelle). Par exemple, toutes les voitures ont un moteur mais toutes les voitures n'ont pas la climatisation et donc la climatisation est une entité fonctionnelle optionnelle qui implique la présence de nouveaux composants au niveau du système de ventilation.

**Notations :**

On fait l'hypothèse qu'il faut désassembler *S* produits de *K* types différents *de la même famille*. Ici, la différence entre les produits de la même famille est due aux années de fabrication différentes, au fabricant et aux options du produit. Chaque produit a un numéro associé. Ce numéro donne l'ordre d'entrée du produit sur la ligne.

On peut avoir deux ou plusieurs types de produits qui entrent alternativement sur la ligne de désassemblage. Par exemple le produit no. *1* est de type $k_1$, le produit *2* est de type $k_2$, le produit *3* est aussi de type $k_1$, ainsi de suite. Dans ce cas on peut construire une fonction surjective qui définit le séquencement donc l'ordre de passage des produits sur la ligne :

$$s : \{1,2,..., S\} \rightarrow \{1,2,..., K\} \qquad (16)$$

La formule (16) devient :

$$f(t) = \sum_{i=1}^{n} \left( H - \sum_{s=1}^{S} \sum_{j=1}^{m_i} t_j^{\,s(s)} \right)^2 \qquad (17)$$

La notation $t_j^{s(s)}$ signifie que les temps opératoires de désassemblage sont dépendants de chaque type ou modèle de produit.

3.2.2.2. Equilibrage en temps réel

Suite aux aléas du processus de désassemblage l'ordonnancement prévisionnel ne peut pas satisfaire le séquencement réel des produits sur la ligne de désassemblage. Il faut donc prendre en considération un ordonnancement dynamique pour corriger les perturbations qui interviennent et pour réordonnancer le flux.

*a. Cas mono produit*

On fait les hypothèses suivantes :

- on a S produits identiques à désassembler
- les temps opératoires sont connus
- l'objectif de la commande est d'établir le type de désassemblage (propre ou destructif) le niveau de désassemblage, la répartition des tâches sur les postes et d'assurer le désassemblage de la totalité des produits dans le temps de travail restant

Pour simplifier la présentation, nous allons placer le moment initial $t$ d'évaluation de la fonction objective pendant la semaine de travail. On note $T_r$ le temps de travail restant :

$$T_r = H - t \qquad (18)$$

On envisage de modifier les formules précédentes en prenant en considération les dernières observations. On va tenir compte maintenant que l'ordonnancement est faite sur le temps restant de désassemblage $T_r$ pendant le fonctionnement de la ligne.
Ainsi l'équation d'équilibrage est :

$$f(t) = \sum_{i=1}^{n} \left(T_r - T_{ri}\right)^2 \qquad (19)$$

où $T_{ri}$ est la charge résiduelle prévisionelle de $W_i$, exprimée en temps de travail.

**Notations :**

Soit $n$ le nombre de postes de travail et $m$ le nombre d'opérations de désassemblage.

Pour évaluer le temps $T_{ri}$ on a introduit trois **variables binaires :**

$j$ - l'indice d'affectations autorisées des tâches sur les postes :

$$j_{ij} = \begin{cases} 1 & si la tâche\ j\ est affectée au\ poste\ i \\ 0 & si la tâche\ j\ n'est\ pas affectée au poste\ i \end{cases}$$

$y$ - l'indice d'état de la tâche qui définit quelles tâches sont déjà accomplies et quelles tâches sont encore à faire :

$$y_j = \begin{cases} 1 & si la tâche j n'est pas encore accomplie \\ 0 & si la tâche j est déjà accomplie \end{cases}$$

$q$ - l'indice qui définit la modalité de réalisation de la tâche :

$$q_j = \begin{cases} 1 & si la tâche j est à faire en mode \textit{propre} \\ 0 & si la tâche j est à faire en mode \textit{destructif} \end{cases}$$

On s'intéresse aux cas suivants :

| Cas | $j_{ij}$ | $y_j$ | $q_j$ | | Discussion |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | ? | l'opération j est faite sur le poste i en mode destructif |
| 2 | 1 | 0 | 1 | ? | l'opération j est faite sur le poste i en mode propre |
| 3 | 1 | 1 | 0 | ? | l'opération j est à faire sur i en mode destructif |
| 4 | 1 | 1 | 1 | ? | l'opération j est à faire sur i en mode propre |

Tableau 1. Les combinaisons possibles des trois indices

Avec ces notations, dans le cas d'une ligne de désassemblage mono produit l'équation (20) devient

$$f(t) = \sum_{i=1}^{n} \left( T_r - S_r \cdot \sum_{j=1}^{m} j_{ij} \cdot y_j \cdot \left( q_j \cdot t_j + (1 - q_j) \cdot t_j^{'} \right) \right)^2 \quad (20)$$

**Remarques :**

- Dans la formule (20) les $t_j$ sont les temps opératoires pour les opérations de désassemblage et les $t_j^{'}$ sont les temps opératoires pour les opérations destructives.
- $S_r$ est le nombre des produits de même type restant à désassembler au moment de l'évaluation de la fonction $f$
- Toutes les variables binaires sont connues au moment de l'évaluation de la fonction objective
- Le temps de travail restant et les temps opératoires sont connus
- La variable binaire $y$ n'est pas dépendante de poste
- L'objectif est de finir le travail dans le temps qui reste et on le fait en minimisant la fonction d'équilibrage d'équation (20)

*b. Cas multi produit*

Les hypothèses sont :

- on a S produits de la même famille à désassembler
- les temps opérationnels sont connus
- le séquencement est établi et ne peut pas être changé
- les opérations à accomplir et leurs relations de précédences peuvent être représentées par un seul graphe de précédence où les noeuds sont les opérations génériques;

La formule (18) devient

$$f(t) = \sum_{i=1}^{n} \left( T_r - \sum_{s}^{S} \sum_{j=1}^{m} j_{ij}^{s(s)} \cdot y_{j}^{s} \cdot \left( q_j^s \cdot t_j^{s(s)} + (1 - q_j^s) \cdot t'^{s(s)}_{j} \right) \right)^2 \quad (21)$$

**Remarques :**

- L'indice d'affectation est dépendant du type de produit, puisque on est parti de la gamme générique mais on doit tenir compte de la présence des sous-ensembles différents entre les produits de la même famille;
- L'indice d'observabilité est individuel due aux aléas du processus de désassemblage qui apparaissent même au cours du développement du désassemblage de deux produits identiques;
- L'indice $q$ est dépendant de no. d'ordre du produit sur la ligne puisque les états individuels des produits en fin de vie imposent un indice individuel; les indices de contrôlabilité sont considérés *des variables de décision*;
- Au moment $t_0$ quand la fonction est évaluée, il y a des produits qui sont sortis de la ligne, donc certains termes de la somme $\sum_{s=1}^{S}$ seront nuls.
- Au moment $t_0$ toutes les variables binaires sont connues.

### 3.3. Evaluation de l'ordonnancement du désassemblage par rapport au critère économique

Le désassemblage induit dans le même temps les coûts et les revenus des sous ensembles ou des composants récupérés par le processus. Ainsi, un bon compromis doit être trouvé qui dépend de la profondeur et de la gamme de désassemblage. Ce problème d'optimisation est dépendant de la structure du système de désassemblage. Si le système est formé d'un seul poste de travail, les coûts sont proportionnels à la durée du processus.

Si le système est une ligne de désassemblage les coûts sont dépendants de l'équilibrage de la ligne quand le système est manuel.

Minimiser les coûts et maximiser les revenus sont des opérations d'optimisation du processus de désassemblage qui prennent en considération d'une part les temps opérationnels pour une affectation donnée des tâches sur les postes et les options de fin de vie des composants et leurs méthodes de valorisation d'autre part. Pour chaque gamme de désassemblage les experts peuvent établir le revenu associé à chaque composant ou sous-ensemble en fonction de sa destination à la fin de vie [Addouche, 2002]. Le profit final $r$ obtenu après le désassemblage d'un produit est la somme des revenus partiels.

$$r = \sum_i r_i \qquad i = 1..m \qquad (22)$$

Où $m$ est le nombre des composants finaux obtenus après le processus de désassemblage.

De même, le coût est dépendant des temps opératoires. D'habitude, l'affectation des tâches sur les postes est faite de telle manière qu'il n'existe pas des différences entre les temps opératoires. Dans telle situation on dit que la ligne est équilibrée. Des temps égaux signifient des coûts égaux des opérations sur les postes. C'est pourquoi les coûts opérationnels sont considérés habituellement proportionnels aux temps opérationnels.

La fonction de coût à minimiser est donnée par l'équation (13). Sa valeur minimale est valide pour une affectation des tâches qui donne un bon équilibrage de la ligne. Minimiser le temps de cycle est minimiser les coûts opérationnels. Tenir compte des affirmations précédentes, on doit trouver une fonction qui mélange les deux : les coûts de désassemblage et les revenus obtenus après la valorisation ou le recyclage des composants. C'est un problème typiquement de multicritère qui maximise la différence ***revenu*** - $l$ ***f*** où $f$ est prise de l'équation (12) et $l$ est un poids. Le problème revient maintenant à calculer $l$ . Pour éviter ce calcul laborieux nous considérons la fonction suivante [Duta, 2003] :

$$f = \frac{r}{t_{cy}} \qquad (23)$$

Dans l'expression de cette fonction $r$ est le revenu pris de l'équation (22) et $t_{cy}$ est le temps de cycle associé à la ligne de désassemblage. Cette fonction représente le flou du revenu dans le système de désassemblage. Un inconvénient de cette fonction est qu'il y a des situations quand un temps de cycle inférieur donne une ligne déséquilibrée. C'est le motif pour lequel on devrait considérer la meilleure affectation des tâches qui donne un bon équilibrage de la ligne et alors calculer la valeur maximale de la fonction F qui prend en considération la valeur du temps du cycle obtenue pour une ligne équilibrée.

Cette fonction est bien évidemment la bonne fonction à optimiser parce qu'elle tend vers la minimisation du temps de cycle dans un système de désassemblage manuel autant qu'automatique. La formule (23) a l'avantage de prendre en considération les coûts opératoires (proportionnels avec les temps opératoires) et les revenus obtenus après le démontage. Maximiser la fonction F de l'équation (23) n'induit pas l'équilibrage optimal de la ligne mais donne une solution qui améliore cet équilibrage. L'objectif de la commande est d'assurer le désassemblage des produits dans la période restante en maintenant le meilleur équilibrage de la ligne, ce qui permet d'obtenir un processus efficace par rapport du critère économique. Le problème d'optimisation est multicritère. Une somme pondérée de termes dépendants de l'équilibrage et du revenu comme fonction objectif est idéale. Malheureusement, c'est difficile d'établir le poids réel entre les deux critères.

### 3.4. Conclusions

Au niveau de l'ordonnancement de la ligne de désassemblage la première étape est d'affecter les opérations à chaque poste de travail, donc d'établir l'ordre des tâches sur les postes de telle manière qu'il n'y ait pas de temps morts sur un poste ou autre. Le résultat de cette étape s'obtient par l'ordonnancement hors ligne.

Cependant, dans un processus de désassemblage en vue de recyclage, le revenu obtenu par la récupération des matières ou composants est un paramètre à maximiser. Ce paramètre est évalué par rapport à la "profondeur" du désassemblage autrement dit en fonction du profit obtenu par la valorisation des constituants récupérés (composants élémentaires ou sous-ensembles). Une optimisation multicritère s'impose. La fonction proposée sera analysée et optimisée dans le chapitre suivant.

La deuxième étape consiste à maintenir l'équilibrage de la ligne. On est intéressé par l'équilibrage en temps réel. Cet objectif est plus difficile à atteindre puisque dans le désassemblage les temps opératoires ont une certaine dispersion. Les temps opératoires du désassemblage propre sont différents de ceux du démantèlement. De plus, les temps opératoires ont une valeur égale à zéro pour les opérations qui manquent dans certains types de produits quand les options de fabrication sont prises en compte. Pour atteindre cet objectif on a introduit trois variables binaires : deux variables d'état et une variable de décision. Les deux premières sont considérées variables d'entrée et la dernière est une variable de sortie par rapport au problème d'ordonnancement en temps réel. Tenant compte des ces valeurs la commande du système va assurer le traitement complet des produits à désassembler dans le temps de travail restant et va garder un bon équilibrage de la ligne.

# CHAPITRE IV

# ALGORITHMES D'OPTIMISATION APPLIQUÉS À L'ORDONNANCEMENT DU DÉSASSEMBLAGE

## 4.1. L'ordonnancement et l'optimisation combinatoire

Le problème d'ordonnancement fait partie des problèmes d'affectation généraux qui représentent une classe importante des problèmes d'optimisation combinatoire. L'optimisation combinatoire est un domaine qui mélange les connaissances des mathématiques appliquées, de la recherche opérationnelle et de l'informatique pour résoudre les problèmes d'optimisation des processus discrets. Trouver une solution optimale pour le problème d'ordonnancement est difficile car le problème est NP complet. Pour résoudre le problème en manière exacte la seule méthode est de faire une énumération complète de toutes les solutions admissibles. Ce processus prend du temps. Le recherche d'une solution optimale peut être totalement inappropriée dans certaines applications pratiques en raison de la dimension du problème et de la dynamique qui caractérise l'ènvironnement du travail. Lorsqu'elle est applicable, une méthode exacte est souvent beaucoup plus lente qu'une méthode approchée. De même, le manque de précision dans la récolte des données, la difficulté de formuler les contraintes ou les objectifs contradictoires peuvent retarder l'obtention d'une solution optimale. Ainsi, il est nécessaire de trouver un mode de résolution qui fournisse une bonne solution dans un temps raisonnable, c'est pourquoi il est fréquent de traiter ce type de problème par des méthodes appelées meta heuristiques, qui se contentent généralement de solutions approchées. C'est le cas de l'ordonnancement du désassemblage McGovern et Gupta ont démontré que le problème d'équilibrage d'une ligne de désassemblage est un problème d'optimisation combinatoire [McGovern, 2003]. Pour positionner notre recherche par rapport aux travaux des autres chercheurs, nous allons présenter brièvement quelques méthodes d'optimisation combinatoire dans les paragraphes suivantes.

## 4.2. Méthodes de résolution des problèmes d'ordonnancement

Les problèmes d'ordonnancement sont très varié s ce qui necessite une grande diversité de méthodes de résolution. La complexité croissante des problèmes d'ordonnancement est un défi pour les chercheurs qui utilisent les méthodes d'optimisation combinatoire pour les résoudre. Dans la littérature il y a essentiellement deux techniques qui assurent

la résolution des problèmes d'ordonnancement : l'utilisation des méthodes exactes (tel que le back tracking, l'algorithme de Djikstra, branch and bound, divide et impera, programmation dynamique, programmation linéaire, etc) et l'utilisation des méthodes approchées (les méthodes meta heuristiques ou les algorithmes évolutifs).

### 4.2.1. Les méthodes exactes

Sont généralement utilisées pour résoudre les problèmes de petite taille pour lesquelles le nombre de combinaisons possibles est suffisamment faible pour pouvoir explorer l'espace de solutions dans un temps raisonnable. Par exemple *la technique de séparation et d'évaluation* (connue sous le nom de *branch and bound*) permet l'exploration de l'espace de recherche par la construction d'une structure arborescente du problème qui se décompose en sous problèmes. Cette arborescence est explorée de façon à éviter les branches ne contenant pas des solutions réalisables et les branches n'amenant pas à des solutions meilleures que la solution courante. Cette méthode a été utilisée la première fois par Johnson en 1981 pour résoudre le problème d'équilibrage des lignes d'assemblage [Johnson, 1981].

*La programmation dynamique* est une méthode d'optimisation opérant par séquences. Son efficacité repose sur le principe d'optimalité de Bellman. Ce principe permet une résolution ascendante qui détermine une solution optimale d'un problème à partir des solutions de ses sous problèmes. Cette méthode nécessite une formulation du critère sous forme d'une relation de récurrence entre deux niveaux successifs. La gamme de problèmes à résoudre à l'aide de cette méthode est plus large que celle de la méthode présentée précédemment. Par contre, la taille du problème est plus limitée. La première procédure de programmation dynamique a été utilisée par Jackson en 1956 pour résoudre le problème simple d'équilibrage de la ligne d'assemblage [Jackson, 1956]. Dans l'approche de Jackson les stations sont représentées par les noeuds d'un graphe et les charges par des arcs étiquetés par des temps d'inactivité et le problème SALBP-1 devient le problème de le plus court chemin. La programmation dynamique a été utilisée par Agnetis et son équipe dans [Agnetis, 1995] pour équilibrer une ligne d'assemblage pour les voitures. La solution est donnée dans un temps polynomial. Gungor et Gupta ont utilisé un algorithme similaire pour trouver la solution du problème d'équilibrage des lignes de désassemblage dans [Gungor, 2001]. L'algorithme de Dijkstra a été utilisé pour déterminer le plus court chemin dans un graphe de précédence pour minimiser le temps d'inactivité de chaque station de désassemblage.

*La programmation linéaire* est l'une des techniques classiques de la recherche opérationnelle. Elle permet la modélisation d'un problème d'optimisation d'une fonction objectif de plusieurs variables en présence de contraintes sous la forme d'un programme mathématique. Si la fonction et les contraintes sont toutes des combinaisons linéaires de variables le programme est dit linéaire. Dans [Addouche, 2002] l'auteur a proposé la programmation linéaire pour trouver la gamme optimale de désassemblage mais il ne l'utilise pas dans l'équilibrage de la ligne de désassemblage.

### 4.2.2. Les méthodes approchées

Une solution optimale est obtenue après la recherche complète de l'espace des solutions. Malgré l'évolution de l'informatique et des méthodes mathématiques il y a des problèmes d'optimisation avec une taille prohibitive de l'espace de solutions admissibles. Dans de nombreuses applications réelles il est impossible de trouver la solution optimale en raison de la dynamique du système étudié, des contraints et du nombre de variables. Compte tenu de ces difficultés, la plupart des spécialistes utilisent les techniques approchées de recherche. L'utilisation d'une méthode approchée d'optimisation ne garantit pas une solution optimale mais une bonne solution dans un temps de calcul raisonnable.

Parmi les techniques approchées de recherche *les méthodes heuristiques* exploitent le mieux le problème à optimiser. Elles produisent une solution non nécessairement optimale mais admissible dans un temps polynomial. De plus, les méthodes heuristiques peuvent être combinées avec d'autres méthodes d'optimisation. En [Gungor, 1999b] les auteurs ont utilisé une méthode heuristique pour résoudre le problème d'équilibrage d'une ligne de désassemblage pour les ordinateurs. Une approche systémique a été utilisée pour mettre en évidence les caractéristiques uniques du processus de désassemblage. Les critères pris en considération ont été : la quantité demandée de chaque sous-ensemble, les temps opératoires de désassemblage, les directions de désassemblage et les relations de précédence entre les composants. Des fonctions de priorité ont été définies pour chaque critère.

*Les techniques de recherche locale* sont des méthodes approchées itératives qui explorent l'espace d'états en partant d'une solution admissible choisie à l'aide d'une heuristique. La recherche s'arrête après un nombre d'itérations ou quand la solution courante ne s'améliore plus. Le recuit simulé, la méthode Tabou, la méthode de descente sont des techniques de recherche locale qui seront présentées brièvement dans ce qui suit.

*Le recuit simulé* est inspiré par l'étude de la stabilité thermique d'un système physique. La méthode part d'une solution initiale admissible et continue l'exploration de l'espace d'états en effectuant des perturbations mineures sur la solution courante. Si la nouvelle solution obtenue est améliorée alors elle est retenue. Si elle est détériorée par rapport au critère d'optimisation alors elle est retenue avec une probabilité inversement proportionnelle au nombre d'itérations. Le recuit simulé a l'avantage de couvrir un espace de recherche plus grand et d'éviter la convergence prématurée vers un optimum local. Plusieurs problèmes de job shop ont été traités par la méthode de recuit simulé. .

*La méthode Tabou* est basée sur deux principes. Le premier est d'améliorer à chaque itération la valeur de la fonction objectif en choissant à chaque fois la meilleure solution voisine. Si celle-ci n'existe pas, le choix se fait sur le moins mauvais des voisins. Le deuxième principe consiste à garder en mémoire les dernières solutions choisies et de ne plus les prendre en considération. L'inconvénient de cette méthode est que si la solution atteinte est tout proche de l'optimum global il y a la possibilité de ne pouvoir pas

échapper à l'optimum local ateint. La méthode Tabu a été utilisée dans l'ordonnancement des lignes flow shop ou job shop.

*La méthode de descente* est une autre méthode de recherche locale. Cette méthode explore l'espace en choisissant chaque fois la meilleure solution voisine de la solution courante. Le procédé continue aussi longtemps que la valeur de la fonction objective diminue. Quand l'optimum local est atteint la descente s'arrête. La méthode est utilisée avec succès dans les problèmes d'optimisation combinatoire qui demandent une solution rapide. L'inconvénient de cette méthode est qu'elle s'arrête après un optimum local. Pour faire face à ces carences, d'autres méthodes de recherche locale plus sophistiquées ont été développées. Un exemple sera donné plus tard dans ce chapitre.

### 4.2.3. Les méthodes évolutives

Les méthodes évolutives sont basées sur des algorithmes inspirés des phénomènes réels. Dans cette catégorie entre les algorithmes génétiques et l'algorithmes de fourmis. Contrairement aux méthodes exactes ou approchées qui donnent une solution unique, les méthodes évolutives traitent un ensemble de solutions admissibles. La méthode de colonisation ou l'algorithme de fourmi est un algorithme probabiliste révolutionnaire qui utilise des agents nommés "fourmis". Les fourmis sont placées dans les noeuds d'un graphe et suivant l'algorithme elles vont trouver le plus court chemin vers la source de la nourriture. McGovern et Gupta ont appliqué l'algorithme des fourmis au problème de l'équilibrage de la ligne de désassemblage [McGovern, 2004]. A chaque partie désassemblée une fourmi est associée. De même, une probabilité de visiter la station de travail est associée à chaque fourmi. L'algorithme de McGovern and Gupta minimise le temps de cycle de la ligne par la minimisation des temps d'inactivité.

*Les algorithmes génétiques* sont, à l'heure actuelle, l'approche la plus utilisée parmi les méthodes évolutives. Les solutions sont codées par une structure chromosomiale. Des opérateurs spécifiques sont appliqués à une population d'individus (équivalent à l'ensemble de solutions admissibles) pour obtenir une autre population avec des performances supérieurs. Une méthode évolutive tente d'améliorer la qualité de la population courante par des réplications, mutations et des sélections. Les algorithmes génétiques permettent de trouver une bonne solution dans un temps réduit de calcul. L'évolution de la population permet de chercher tout l'espace des solutions. De plus, les algorithmes génétiques sont appropriés à l'optimisation multicritère. L'équipe du professeur Gupta utilisé des algorithmes évolutifs dans [McGovern, 2004]. Le chromosome est une affectation possible des tâches aux postes de travail. Pour obtenir l'affectation optimale et l'équilibrage de la ligne, les opérations de mutation et séléction ont été apliquées aux nouveaux chromosomes obtenus. L'application des algorithmes génétiques a donné une très bonne solution au problème d'équilibrage pour des problèmes comportant jusqu'à 20 tâches de désassemblage. Au dessus de cette valeur le temps de calcul augmente la méthode des algorithmes génétiques génère environ 100000 populations et donc cette méthode n'est plus efficace.

## 4.3. Application d'une méthode exacte à l'ordonnancement du désassemblage

### 4.3.1. La description de la méthode [Duta, 2003a]

Reprenons la fonction de coût dans le système de désassemblage de l'équation (23) :

$$f = \frac{r}{t_{cy}}$$

Pour obtenir la valeur minimale de cette fonction nous allons appliquer un algorithme classique à l'aide de la *technique backtraking.* On fait les hypothèses suivantes :

***Hypothèses :***

1. L'architecture de la ligne est de type flow shop
2. Les postes peuvent accomplir des opérations de désassemblage destructives ainsi que non-destructives.
3. Les temps de désassemblage sont des temps moyens estimés
4. Les gammes de désassemblage sont données dans le réseau de Petri.

***Notations :***
m : le nombre des opérations de désassemblage
n : le nombre des postes de travail
F : la fonction d'équilibrage de l'équation (13)
$t_{cy(k)}$ : le temps de cycle pour la gamme k
$t_j$ : le temps opératoire de l'opération j
$T_k$ : le vecteur des temps opératoires pour la gamme k
$T$ : la matrice des vecteurs $T_k$
R : le vecteur des revenus finaux $R^k$ pour la gamme k
$M = \{m_{ij}\}, i = 1..n,\ j = 1..m$ : une matrice binaire d'affectation possible, où

$$m_{ij} = \begin{cases} 1 \ sil'opération\ j\ peut\ \hat{e}tre assignée au poste i \\ 0 \qquad\qquad\qquad\qquad\qquad encas contraire \end{cases}$$

$S = \{s_{ij}\}, i = 1..n,\ j = 1..m$ : la matrice solution (affectation des opérations sur les postes)
avec les éléments $s_{ij} = \begin{cases} 1 \ sil'opération j est assignée au\ poste i \\ 0 \qquad\qquad\qquad\qquad\qquad encas contraire \end{cases}$

La matrice S est soumise aux contraintes suivantes :

$$s_{ij} \in \{0,1\} \qquad (24)$$

$$\sum_i s_{ij} = 1 \qquad (25)$$

$$\sum_{k=1}^{n} k \cdot s_{ki} - \sum_{k=1}^{n} k \cdot s_{kj} \leq 0 \qquad (26)$$

La relation (24) s'appelle **contrainte de non divisibilité** et représente la condition que une opération ne peut pas être divisée entre les postes de travail.

La relation (25) est la **contrainte d'affectation** qui demande que l'opération soit assignée à un seul poste.

L'inegalité (26) est la **contrainte de précédence.** Elle trasuit l'ordre technologique des opérations : si l'opération i doit être réalisée avant l'opération j, alors l'opération i ne peut pas être assignée au poste $P_i$ placé en aval du poste $P_j$. L'objectif de la méthode proposée est de trouver les matrices S qui satisfont les trois contraintes.

Les matrices S sont déterminées par une procédure classique qui calcule toutes les affectations possibles à l'aide de la méthode de programmation backtracking appliquée au produit cartésien. Le nombre des éléments du produit cartésien est $p$ calculé par la formule $p = \prod_{j=1}^{m}\left(\sum_{i=1}^{n} m_{ij}\right)$.

L'algorithme ci-après utilise une procédure **Génerer_S** qui génère toutes les matrices S :

**DEBUT**
**Etape_1**
Lire les données d'entrée *m, n, M, T,* et *R.*

**Etape_2**
Générer le produit cartésien

$$A_1 \times \ldots \times A_m = \left\{ \left(e_1, \ldots, e_m\right)_1 \ldots \left(e_1, \ldots, e_m\right)_p \mid e_i \in A_i, i = 1..m \right\}$$

Où $A_i$ = l'ensemble des indices des postes qui peuvent accomplir l'opération *i* et *p* le nombre d'éléments;

**Etape_3**
Pour chaque $v = 1..p$
*Début*
Générer_$S_v$ (n, m, $e_i$);
Pour $S_v$ générée
*Début*
Verifier les rélations (25) and (26) and (27);
Si les trois contraintes sont vérifiées alors
Mémoriser la matrice $S_v$ ;
*Fin;*
*Fin;*

**Etape_4**
Pour chaque gamme *k*
Pour chaque matrice $S_v$ determinée à l'Etape_*3*
*Début*

$$Q_{vk} = S_v \cdot T_k = \{q_i\}_{vk} \overset{not}{=} q_{i(vk)}$$

$$t_{cy(k)} = \max_{W_i} \sum_{j \in (tasks W_i)} t_{j(k)}$$

$$F_k = \sum_{i=1}^{n} \left(q_{i(vk)} - t_{cy(k)}\right)^2$$

*Fin.*

**Etape_5**

Calculer le $\min_k\{F_k\}$ et trouver la matrice $S_{opt}$ qui minimise cette fonction.

**Etape_6**

Pour $S_{opt}$ calculée à l'Etape_5
Pour chaque gamme $k$ calcule

$$\max_k f_k = \frac{R^k}{t_{cy(k)}}$$

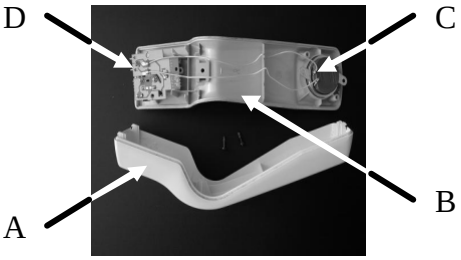
**Etape_7**

Afficher le no $k$ de la gamme qui maximise la fonction f à l'Etape_6;
Afficher dans un ordre décroissante les valeurs de la fonction $f_k$

**FIN.**

### 4.3.2. Application de la méthode exacte

L'exemple utilisé est le désassemblage d'un récepteur téléphonique [Zussman, 1999].
Le réseau de Petri est donné dans la figure 21. Un revenu $r$ est associé à chaque position et un coût à chaque transition. Le coût est proportionel avec les temps opératoires $t_j$.




A carcasse de plastique

B carcasse de plastique

C diffuseur d'acier

D plaque circuit imprimé

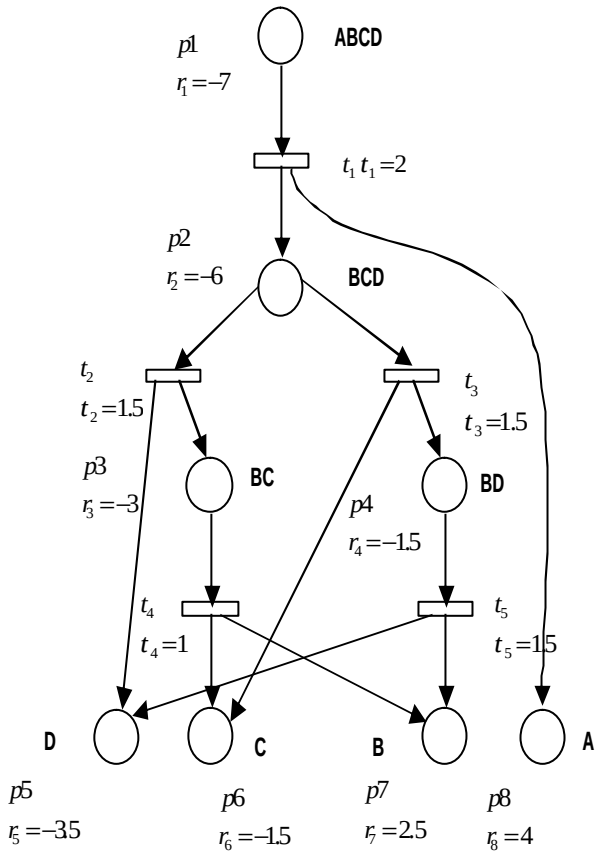



Fig. 21. Le réseau de Petri pour un récepteur téléphonique simple

Il y a deux gammes principales de désassemblage propre et elles sont décrites en ordre par la séquence des transitions franchies :

$t_1 \to t_3 \to t_5$ $(k = 1; gamme 1)$

$t_1 \to t_2 \to t_4$ $(k = 2; gamme\ 2)$

On fait l'hypothèse que la ligne de désassemblage a deux postes de travail, un poste pour les opérations destructives et l'autre pour le désassemblage propre.

On a pris en considération l'échec du désassemblage propre. Par exemple le $t_2^{'}$ et le $t_5^{'}$ sont les temps opératoires pour les opérations d'extraction ou découpage du composant D. Alors, les gammes destructives sont representées par les transitions :

$t_1 \to t_3 \to t_5^{'}$ $(k = 3; gamme 3)$

$t_1 \to t_2^{'} \to t_4$ $(k = 4; gamme 4)$

Touefois, il y a les gammes partielles suivantes :

$t_1 \to t_3 \ \ (k = 5);\ t_1 \to t_2 \ \ (k = 6);\ t_1 \ \ (k = 7).$

Le revenu total pour chaque gamme est calculé à l'aide de l'équation (22) :

$R^1 = r_5 + r_6 + r_7 + r_8 = 1.5$ $\qquad R^5 = r_4 + r_6 + r_8 = 1$

$R^2 = r_5 + r_6 + r_7 + r_8 = 1.5$ $\qquad R^6 = r_3 + r_5 + r_8 = -2.5$

$R^3 = r_5^{'} + r_6 + r_7 + r_8 = 1$ $\qquad R^7 = r_2 + r_8 = -2$

$R^4 = r_5^{'} + r_6 + r_7 + r_8 = 1$

où $r_5^{'} < r_5$ est le revenu obtenu par une opération destructive.

Les données :

Le nombre d'opérations possibles m=5, le nombre de postes n=2, le nombre des gammes possibles d=7;

Les matrices sont données ci-aprés:

$$M = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$R = \{R^k\} \qquad k = 1..7$

$$T = \begin{pmatrix} 2 & 0 & 1.5 & 0 & 1.5 \\ 2 & 1.5 & 0 & 1 & 0 \\ 2 & 0 & 1.5 & 0 & 1 \\ 2 & 1 & 0 & 1 & 0 \\ 2 & 0 & 1.5 & 0 & 0 \\ 2 & 1.5 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \end{pmatrix}$$

L'algorithme donne comme résultat *la gamme optimale, l'affectation optimale et la valeur maximale de la fonction objective.* La fonction de coût est exprimée en euros par seconde.

$$k=2;\ S_2 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \text{ et } \max_k f_k = f_2 = 0.6$$

Ainsi, la gamme optimale qui maximise la valeur de la fonction f de l'équation (24) est la gamme répresentée par la succession des transitions :

$$t_1 \rightarrow t_2 \rightarrow t_4$$

L'affectation des opérations : $Op_1$ au poste $W_1$, $Op_2$ et $t_4$ au poste $W_2$. Toutefois, la ligne est bien équilibrée, la valeur de la fonction d'équilibrage de l'équation (12) est minimisée.

Les valeurs obtenues pour la fonction objectif *f* à chaque gamme sont les suivantes :

$$f_k = \{0.5;0.6;0.4;0.5;0.5;-1.25;-1\}$$

Si le désassemblage total n'est pas possible, le désassemblage partiel est réalisé. A la suite de l'analyse des résultats, la conclusion est que la gamme partielle $t_1 \rightarrow t_3$ est plus profitable que la gamme partielle $t_1 \rightarrow t_2$. C'est à dire que c'est plus profitable de séparer C du sous-ensemble BD que D du BC. De plus, les valeurs de la fonction objectif montrent qu'il y a des gammes destructives préférées à la place d'une gamme partielle non destructive (gammes 3 et 4 préférées à la place du gamme 6).

### 4.4. Application des méthodes stochastiques à l'ordonnancement du désassemblage

#### 4.4.1. L'algorithme Kangourou

4.4.1.1. Les étapes de l'algorithme

Dans le but d'étudier les problèmes d'ordonnancement d'un processus réel de désassemblage nous allons présenter un algorithme stochastique d'ordonnancement inspiré du recuit simulé. L'algorithme s'appelle *algorithme du Kangourou* [Fleury, 1995] qui permet une descente stochastique associée à un système de voisinage de l'état actuel. La méthode est un algorithme itératif qui explore l'espace des solutions en choisissant à chaque fois la meilleure solution voisine de la solution courante. La recherche de la meilleure solution voisine est un problème qui peut être aussi difficile que le problème initial. Le processus de descente s'arrête après un nombre d'itérations fixé au début. La méthode fournit un minimum local.

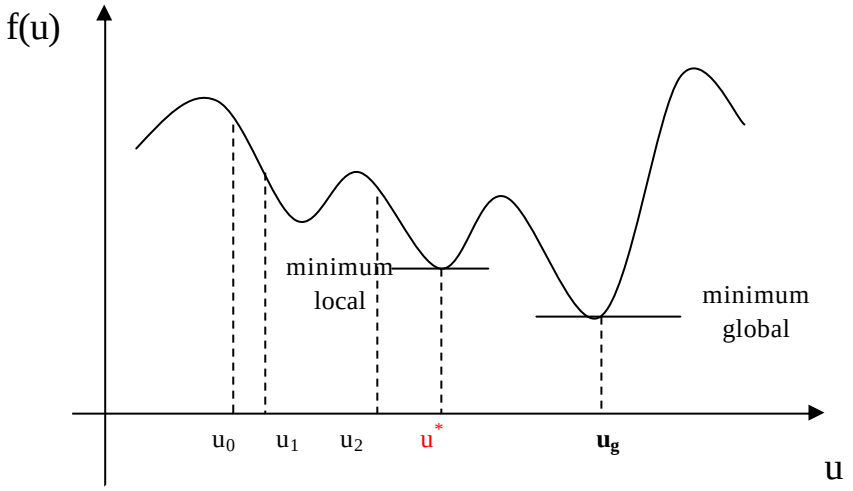



Fig. 22. La descente pour trouver un minimum local

Soit $u_0$ une solution admissible du problème d'optimisation. Par des déplacements succesifs l'algorithme de Kangourou cherche une solution qui minimise la fonction *f* dans un voisinage de la solution courante. Si la solution $u_1$ est meilleure que la solution précedente, elle est memorisée et une nouvelle solution est cherchée dans le même voisinage. Si la solution $u_1$ n'est pas meilleure que la solution précedente, l'algorithme trouve un autre voisinage par un saut. Après un nombre d'itérations un minimum local $u^*$ est trouvé. Ce minimum est plus ou moins proche du minimum global (figure 22). Dans le cas idéal le minimum local $u^*$ est le même avec le minimum global $u_g$.

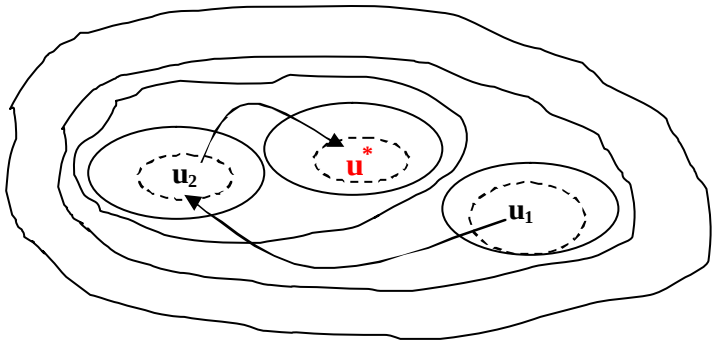



Fig. 23. La recherche d'un minimum local dans le voisinage de la solution courante

L'algorithme de Kangourou offre une solution par une descente stochastique et une transition dans le voisinage de l'état actuel pour trouver une meilleure solution de la solution courante. La méthode donne un optimum local dans un temps acceptable. L'algorithme de Kangourou a été déjà utilisé par V. Minzu dans le cas des lignes d'assemblage pour minimiser le temps de cycle et il a obtenu une bonne solution dans un temps raisonnable.

Dans le paragraphe qui suit nous allons présenter l'algorithme kangourou. L'algorithme prend en considération toutes les affectations initiales possibles des tâches. Nous avons adapté l'algorithme pour le processus de désassemblage.

**Notations :**

u : état courant (partition des tâches),
u* : meilleur état rencontré à l'itération courante,
c : compteur d'itérations entre deux améliorations de la solution,
A : le nombre maximal d'itérations sans l'amélioration de la solution courante (le no. de stationnements avant un saut);
$f$ : la fonction objectif;
m : le nombre des opérations de désassemblage
n : le nombre des postes de travail

**START Algorithme Kangourou**

**Etape_1**
On fixe une valeur $A = a \cdot \left[\frac{m}{n}\right]$ , où $a \in \{1,2\}$,

**Etape_2**
On choisit une solution valide (un état initial) $u$

**Etape_3**
Une meilleure solution $u^*$ est recherchée en vue de minimiser la fonction objective $f(u)$

Soit
$c \leftarrow 1$ ; u* ← u;

**Etape_4**
Répéter
Si c<A alors *descente* (u, u*,c)
Sinon *saut* (u, u*, c);
Jusqu'à l'accomplissement d'un critère établi ;

**FIN Algorithme Kangourou**.

Les deux procédures descente et saut sont décrites ci-aprés :

**Procédure** *descente* (u, u*, c)
Début /* Génération d'une solution v dans le voisinage de u*/
c ← c+1;
Si $f(v) \le f(u)$ alors
Début
Si $f(v) < f(u)$ alors
Début
c ← 0; /* une meilleure solution de la fonction est obtenue;*/
si f(v)<f(u*) alors u* ← v; /* un minimum de la fonction objectif a été obtenu*/
Fin;
u ← v;/* la nouvelle voisinage remplace la vieille */
Fin;
Fin;

```
Procédure saut (u, u*, c)
Début
/* Génération de v dans une voisinage de u*/
c←c+1;
Si f(v)≠f(u) alors
      Début
         Si   f(v)<f(u*)   alors
         u*←v;
            c←0;
         Fin;
         u←v;
Fin.
```

Le premier pas dans l'application d'un algorithme stochastique est la définition de l'espace des solutions et du voisinage dans cet espace. Pour trouver une solution pour notre problème d'optimisation donné par l'équation d'équilibrage (22) il faut définir une répartition valide des opérations sur les postes de travail à l'aide des indices $j_{ij}$ et $q_{ij}$ définis dans le chapitre 3. Une répartition des tâches dans le voisinage de la partition initiale est obtenue par une ou plusieurs des opérations suivantes :

1. Déplacer une opération d'un poste à l'autre
2. Ajouter une nouvelle opération pour augumenter le niveau de désassemblage ou enlever une opération de désassemblage pour réduire le niveau de désassemblage
3. Transformer une opération non-destructive en une opération destructive et inversement

4.4.1.2. L'application de l'algorithme Kangourou

*Exemple 1*. Le désassemblage d'un poste de radio Motorola

Pour illustrer l'algorithme proposé on a pris en considération l'exemple simple du désassemblage d'un poste de radio Motorola donné dans [Salomonski, 1999].

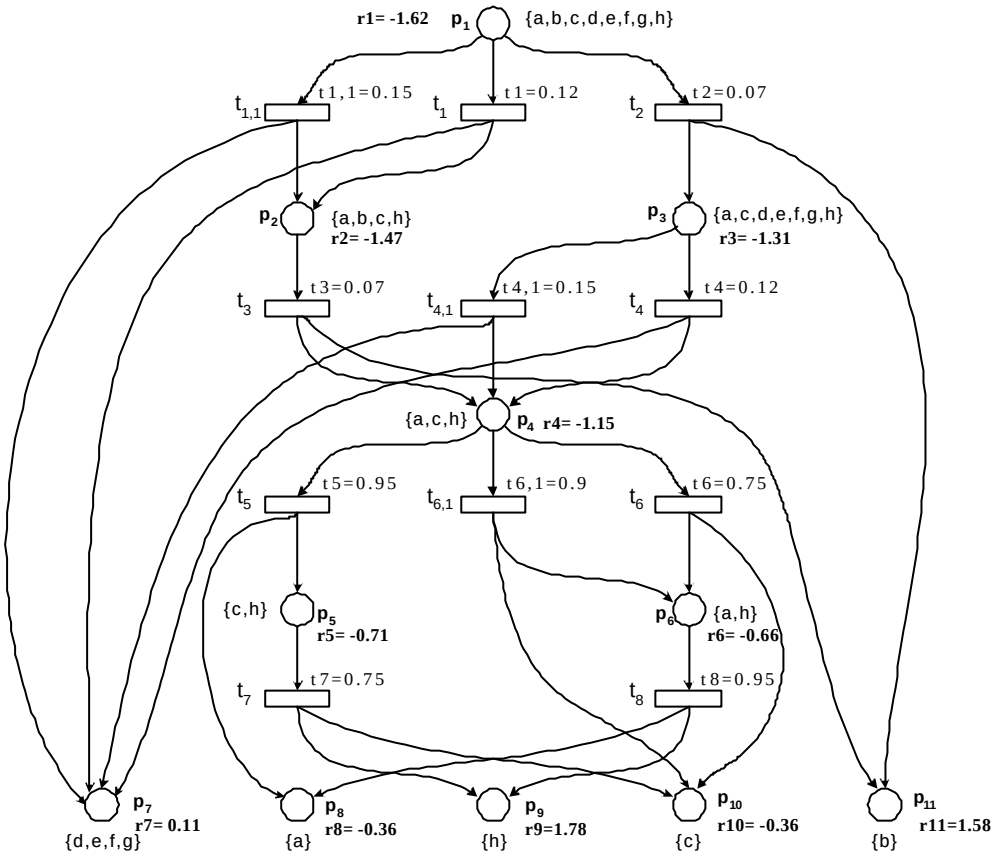



Fig. 24.

Le reseau de Petri de désassemblage d'un radio Motorola (après [Salomonski, 1999])

Le processus de désassemblage est répresenté par quatre gammes de désassemblage possibles : $\{t_1 \to t_3 \to t_5 \to t_7\}$ , $\{t_1 \to t_3 \to t_6 \to t_8\}$ , $\{t_2 \to t_4 \to t_5 \to t_7\}$ et $\{t_2 \to t_4 \to t_6 \to t_8\}$ où les $t_i$ sont les transitions du réseau de Petri associé au produit, transitions equivalentes à l'accomplisment des opérations de désassemblage correspondantes.

On définit les listes suivantes :

$L_{op}$ – la liste des opérations de désassemblage

$L_i$ – la liste d'affectations initiales des opérations aux postes de travail;

$L_f$ – la liste finale des postes sur lesquelles les opérations peuvent être déplacées

Pour prendre en considération les opérations destructives, dans le réseau de Petri correspondant il y a trois transitions alternatives : $\{t_{1,1}, t_{4,1}, t_{6,1}\}$.

Nous avons consideré trois postes de travail. Une opération destructive est faite sur une station mixte qui peut accomplir les deux types d'opérations : de désassemblage propre

et déstructif. Ainsi, les postes 2 et 3 sont mixtes. Avec le déplacement d'une opération d'un poste à l'autre le type de désassemblage est changé ainsi que le temps opérationel. Dans ce cas on dit qu'on obtient un voisinage de l'affectation précédente. On fait l'hypothèse que les opérations $t_1$ et $t_4$ peuvent être accomplies dans un mode non destructif sur le premier poste et dans un mode destructif sur le deuxième. L'opératiopn $t_6$ est accomplie dans un mode destructif sur le poste numéro trois.

**Données d'entrée**
Avec les hypothèses précedentes les listes définies sont :

- $L_{op} = \{op_1, op_2, op_3, op_4, op_5, op_6, op_7, op_8\}$
  $L_i = \{W_1, W_1, W_2, W_1, W_2, W_2, W_3, W_3\}$
- $L_f = \{W_2, W_1, W_2, W_2, W_2, W_3, W_3, W_3\}$

On a consideré A= 4. Le critère d'arrêt est d'accomplir les 10 itérations.
De l'article de [Salomonski, 1999] nous avons pris les durées d'opérations et nous avons obtenu deux vecteurs : $T_i$ - le vecteur des temps opératoires pour les opérations de désassemblage, $T_f$ - le vecteur des temps opératoires dans le cas des opérations destructives.
Alors :

$$\mathrm{T}_i = \{0.12, 0.07, 0.07, 0.12, 0.95, 0.75, 0.75, 0.95\}$$

$$\mathrm{T}_f = \{0.15, 0.07, 0.07, 0.15, 0.95, 0.90, 0.75, 0.95\}$$

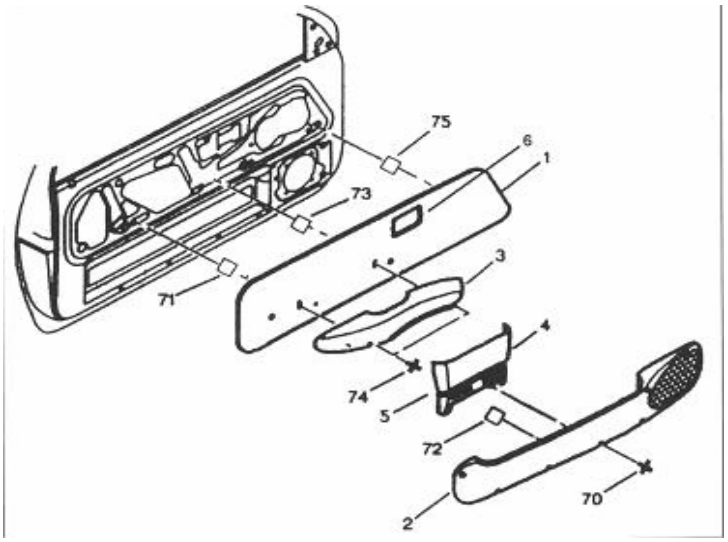
La fonction objectif est donnée par *l'inverse de la fonction (24)* du chapitre précédent parce que on prends en considération la minimisation des coûts opératoires.
Après l'éxécution de l'algorithme Kangourou la valeur maximale de la fonction est
*f(t) = 0.85*

A l'application d'une méthode exacte qui utilise backtracking (l'algorithme de 4.3.1.) sur le même problème on obtient un maximum de 0.92 unitées de valeur/unité de temps .

La gamme optimale est $\{t_2 \to t_4 \to t_{6,1} \to t_8\}$ où la seule opération destructive est $t_{6,1}$.
Toutefois, pour un bon équilibrage l'affectation optimale est $t_2, t_4 \in W_1$ et $t_6, t_8 \in W_3$, la valeur de la fonctio n objectif pour cette configuration est très proche de zéro.
La conclusion est donc que *dans le cas des produits simples les méthodes exactes sont plus appropriées* car le résultat est le minimum global et il est obtenu très rapidement. De plus, le minimul local est rélativement éloigné du minimum global. Pour éviter le blocage dans un optimum local il faut prendre un nombre plus grand d'itérations.

*Exemple 2.* Le désassemblage d'un panneau de porte Peugeot 106

Comme un deuxième exemple on a pris le désassemblage d'une porte du modèle Peugeot 106. Les composants et les temps de désassemblage sont donnés (source [PSA, 1999]).




| | | | |
|---|---|---|---|
| 1 | Paneau garni | 70 | Vis torx |
| 2 | Vide-poche | 71 | Clip de fixation |
| 3 | Accoudoir | 72 | Agrafe |
| 4 | Garniture d'absorbeur | 73 | Ecrou plastique |
| 5 | Absorbeur | 74 | Vis torx |
| 6 | Enjoliveur de poignée<br>Commande manuelle de vitre | 75 | Agrafe |

Fig. 25. Les composants d'une porte Peugeot 106

| **No. d'op.** | **Opération de désassemblage** | **Temps (en s)** |
|---|---|---|
| AM10 | Arracher la commande manuelle de vitre | 4 |
| AM11 | Extraire le vide poche | 89 |
| AM12 | Arracher l'enjoliveur de poignée | 3 |
| AM20 | Arracher la garniture d'absorbeur | 12 |
| AM30 | Extraire l'accoudoir | 21 |
| AM31 | Arracher l'absorbeur | 12 |
| AM40 | Extraire le paneau garni | 41 |
| AM50 | Arracher le textile | 120 |
| AM51 | Arracher les insonorisations | 180 |

Tableau 2. Les opérations principales de désassemblage de la porte

Nous avons ignoré les opérations annexes comme la prise ou le possitionnement d'un outil.

**Hypothèses :**

- il s'agit d'un seul type de produit (Peugeot 106)
- l'horison de planification est H = une semaine
- le nombre de produits de même type à désassembler est constant S=40

- la fonction à optimiser est la fonction d'équilibrage F de l'équation (17) du Chapitre 3
- les temps de désassemblage pour les autres composants sont connus et déterministes
- le temps de cycle est connu et égale à 3600 s pour le désassemblage de la voiture entière
- il y deux postes mixtes où le désassemblage de la porte est réalisé

L'exécution de l'algorithme du [Duta, 2003a] donne la valeure minimale de la fonction *F* de 260 s, ce qui est un bon résultat.

### 4.4.2. Les algorithmes génétiques

4.4.2.1. Présentation sommaire de la terminologie des AG

Comme de nombreuses méthodes de résolution de problèmes d'optimisation, un algorithme génétique part d'un ensemble de solutions acceptables pour améliorer progressivement ses éléments. Un algorithme génétique est un algorithme récursif s'appliquant sur une population de solutions qui génère une génération suivante par croisement dont la composition est modifiée par mutation et sélection. Pour permettre ces transformations, les états solutions doivent subir un codage.

*Un codage informatique* d'une solution est représenté par une structure des données. L'organisation des données dans cette structure influe autant que les valeurs prises par chacune de ses parties. Lorsqu'il s'agit d'une modélisation informatique, on pense à un codage binaire. *Un individu* correspond au codage sous forme de gènes d'une solution potentielle à un problème d'optimisation. Dans un codage binaire, *un gène* vaut soit 0, soit 1. Chaque *génotype* représente une solution potentielle à un problème d'optimisation. La valeur de cette solution potentielle est appelée *le phénotype*. Pour faire naître une nouvelle génération de solutions, remplaçant l'ancienne dans l'ensemble, deux transformations principales seront appliquées avec un ordre qui fait partie des choix de modélisation. Ce sont *le croisement* et *la mutation* [Goldberg, 1994].

*Le croisement* est la clef de la puissance des algorithmes génétiques. Le croisement permet de réunir en un individu fils (ou dans une solution fille) des caractéristiques intéressantes appartenant à un seul de ses parents avec d'autres, présentes chez l'autre parent. C'est une recomposition des caractéristiques de l'ensemble des solutions mais aucune information nouvelle n'est créé, tout provient d'un héritage. Cependant ce phénomène n'est pas totalement déterministe, la méthode d'attribution à la nouvelle génération étant stochastique.

A l'inverse du croisement, *la mutation* s'applique à créer de toute pièce des données, si possible non présentes dans la population parentale. Ce type de modification est, souvent, totalement stochastique. On tire au sort l'endroit où se produit la mutation et la nouvelle valeur. Le nombre de types de mutations différentes est fonction de l'imagination du chercheur. Cependant on en retrouve certaines pour des classes de problèmes comme par exemple la transposition qui consiste à échanger deux gènes dans

un même chromosome, ce qui revient par exemple pour un problème d'ordonnancement à permuter deux sommets ou tâches. Le caractère stochastique de ces opérations induit des contraintes dans leur application. Ainsi, on doit les appliquer sur une population statistique, c'est à dire sur des effectifs importants pour que la diversité des possibles s'exprime. Après la mutation, la population fille contient des solutions ni mieux ni moins bien adaptées en moyenne que la population parentale. Il s'agit, donc d'exercer une *sélection* pour permettre aux "bonnes" solutions de se développer plus facilement que les autres, et pour garder une diversité qui fait la richesse de ce type d'algorithme. Les méthodes de sélection de la solution sont probabilistes. Une des méthodes utilisées est celle dite *de la roulette* : à chaque individu $x_i$ on associe une probabilité d'apparition $p_i$, alors la probabilité de sélection de cet individu est donnée par le rapport $p_i / 100$. Une autre méthode utilisée pour la sélection est *d'évaluer l'individu* par rapport à un critère d'évaluation. D'habitude ce critère est la valeur minimale/maximale de la fonction objective établie dans le problème d'optimisation. Cette valeur s'appelle *le fitness* de la fonction objectif.

Les problèmes d'ordonnancement apportent des contraintes spécifiques pour leur résolution. Dans un graphe de précédence, par exemple, il y a des contraintes de précédence. Ainsi, lorsqu'on code la matrice de précédence comme un individu (chromosome), le croisement et la mutation doivent être faits en respectant ces contraintes. Il convient donc de développer des opérateurs spécifiques pour les opérations de croisement et mutation. De plus, il est souvent intéressant de conserver dans la descendance des propriétés qui s'exprime chez les deux parents. S'il y a précédence entre un sommet A et un sommet B chez les deux parents, cette propriété doit être fortement favorisée dans la solution fille. Ce choix permet de générer des solutions ayant cette propriété dans l'ensemble dé départ pour la conserver par croisement jusqu'a optimum. Pour la mutation, on effectue en général, des transpositions, c'est à dire l'échange de deux gènes soit l'intervertissement de deux sommets (le déplacement des tâches d'un poste à l'autre).

En conclusion, pour pouvoir appliquer un algorithme génétique à la résolution d'un problème d'optimisation, on a besoin :

- d'une représentation binaire de la solution du problème;
- d'une façon pour générer la population initiale, soit aléatoire, soit heuristique;
- d'une fonction d'évaluation;
- des opérateurs génétiques;
- des valeurs pour la dimension de la population;
- des probabilités pour l'application des opérateurs;
- des critères d'arrête (soit le nombre maximal de populations, soit un critère de convergence de la population).

Les phases d'un algorithme génétique sont :

1. Génération de la population initiale;
2. Evaluation par rapport d'une fonction;
3. Sélection

4. Application des opérateurs (de croisement, de mutation, de reproduction, etc.)
5. Remplacement de la population initiale;
6. Répétition des étapes 2-5 jusqu'à la satisfaction d'un critère d'arrêt.

4.4.2.2. Exemple d'application des AG dans l'ordonnancement du désassemblage

L'objectif de l'algorithme est de fournir une solution de désassemblage faisable pour le produit considéré en minimisant les coûts de désassemblage et d'assurer un bon équilibrage de la ligne de désassemblage. L'algorithme a été testé sur l'exemple présenté dans le paragraphe 4.3.2. On garde les mêmes notations. Pour le problème d'ordonnancement on utilise le codage de permutation : dans une matrice (qui represent l'individu) on range les numéros d'opérations dans l'ordre de placement sur les postes. *Le fitness* est donné par la valeur de la fonction objectif pour chaque individu. La fonction objectif de l'équation (23) est évaluée par rapport aux deux premières gammes (le désassemblage complet et propre) pour lesquelles le revenu est maximum. On prend en considération trois postes de travail, cinq opérations de désassemblage possibles (destructives et non-destructives) et aussi les gammes incomplets de désassemblage.

**Début de l'AG**

**Etape 1.** Génération de la population initiale

On part de la matrice d'affectations possibles : $M = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}$

Les solutions de la problème sont représentées par les matrices $S$ qui respectent les contraintes (24), (25), (26). Par la méthode classique de la génération des toutes les solutions possibles décrite dans le paragraphe 4.3.1. On obtient 24 matrices $S$ (la dimension maximale de la population initiale est donnée par la formule $p = \prod_{j=1}^{m} \left( \sum_{i=1}^{n} m_{ij} \right)$). La population initiale est générée en partant de la matrice d'affectations possibles $M$. On part d'une population initiale de trois individus : trois matrices $S$ possibles qui sont soumises aux trois contraintes précédentes. Les dernières deux matrices sont obtenues par le déplacement d'une opération d'un poste à l'autre.

$$S_1 = \begin{pmatrix} 1 & \textcircled{1} & 0 & 0 & 0 \\ 0 & \textcircled{0} & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad S_2 = \begin{pmatrix} 1 & \textcircled{0} & 0 & 0 & 0 \\ 0 & \textcircled{1} & 1 & 1 & \textcircled{1} \\ 0 & 0 & 0 & 0 & \textcircled{0} \end{pmatrix} \qquad S_3 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & \textcircled{0} \\ 0 & 0 & 0 & 0 & \textcircled{1} \end{pmatrix}$$

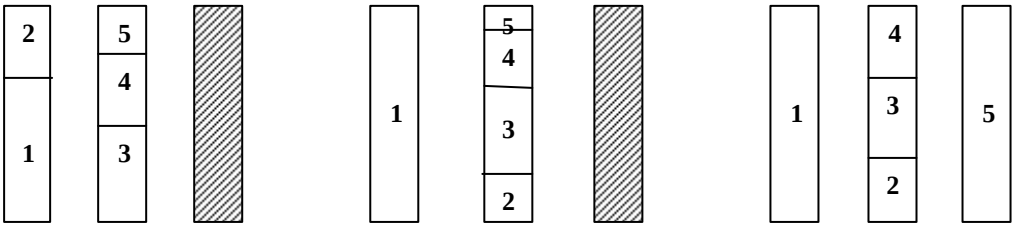


Fig 26. L'affectation initiale correspondante à la population initiale

**Etape 2.** Evaluation par rapport de la fonction objectif

La fonction objectif est la fonction de coût de l'équation (23). Le calcul est fait pour les deux gammes principales (désassemblage propre et comp let).

Tableau 3. L'evaluation de la population initiale

| | | | ***fitness*** | ***robustesse*** | ***moyenne*** |
|---|---|---|---|---|---|
| | $f_1$ | $f_2$ | $(f_1+f_2)/2$ | *fitness %* | |
| $S_1$ | 0.5 | 0.6 | 0.55 | 33% | 0.80 |
| $S_2$ | 0.5 | 0.43 | 0.46 | 27% | 1 |
| $S_3$ | 0.75 | 0.6 | 0.675 | 40% | 1.20 |
| Somme | | | 1.685 | 100% | |
| Moyenne | | | 0.5616 | 33% | |

**Etape 3.** La sélection

Nous avons utilisé la méthode de sélection par le rang de classement. C'est à dire que les individus d'une population sont classées dans l'ordre croissant de leurs fitness. Dans notre exemple, le critère de la sélection est donné par la robustesse d'un individu. Après l'étape de l'évaluation on observe que les plus robustes individus sont le $S_1$ et le $S_3$. On ne peut pas utiliser la méthode de la roulette parce que elle peut générer des individus non- valides qui ne respectent pas les contraintes de précédences.

**Etape 4.** Le croisement et la mutation

*a. Le croisement :* Soit deux matrices de la même dimension

A=$\{a_{ij}\}$ $i=\overline{1..n}$, $j=\overline{1..m}$ et B=$\{b_{ij}\}$ $i=\overline{1..n}$, $j=\overline{1..m}$

On définit *l'opérateur de croisement* comme l'opérateur logique *"or exclusif"* entre les éléments des deux matrices comme il suit :

$$a_{ij} \oplus b_{ij} = \begin{cases} 0 \ si \ (a_{ij}=0 \ et \ b_{ij}=0) \ ou \ (a_{ij}=1 \ et \ b_{ij}=1) \\ 1 \ si \ (a_{ij}=1 \ et \ b_{ij}=0) \ ou \ (a_{ij}=0 \ et \ b_{ij}=1) \end{cases}$$

Soit la matrice masque $MSQ=\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

L'opérateur de croisement est appliqué aux matrices suivantes :

$$S_1 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad S_3 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Par le croisement on calcule $S_1 \oplus MSQ$ et $S_3 \oplus MSQ$ et on obtient les matrices filles suivantes qui ont des gènes des deux parents :

$$S_4 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \qquad S_5 (= S_2) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

On a obtenu un nouvel individu (le $S_4$) et une copie d'un individu (le $S_2$).

*b. La mutation :* *L'opérateur de mutation* est défini par le déplacement d'une opération d'un poste de travail au poste voisin. On applique la mutation par le déplacement d'opération 2 du poste 1 au poste 2 dans le $S_4$ et on obtient un nouvel individu : $S_6 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$

**Etape 5.** Remplacement de la population initiale

La nouvelle population et ses performances sont représentées dans le tableau 4 :

Tableau 4. L'évaluation de la nouvelle population

| | | | ***fitness*** | ***robustesse*** | ***moyenne*** |
|---|---|---|---|---|---|
| | $f_1$ | $f_2$ | $(f_1+f_2)/2$ | *fitness %* | |
| $S_3$ | 0.75 | 0.6 | 0.675 | 36% | 1.09 |
| $S_4$ | 0.75 | 0.43 | 0.59 | 32% | 0.96 |
| $S_6$ | 0.75 | 0.43 | 0.59 | 32% | 0.96 |
| Somme | | | 1.855 | 100% | |
| Moyenne | | | 0.618 | 33% | |

Etape 6. Répétition des étapes 2-5 jusqu'à la satisfaction d'un critère d'arrêt

On répète les étapes 2-5 pour revigorer chaque fois la nouvelle population obtenue. Puisque ce type d'AG appliqué dans le désassemblage garde chaque fois les fils les plus robustes de la population, il y a la possibilité qu'on obtient des duplicates des matrices S et la solution soit attrapée dans un optimum local. Dans l'exemple précédent, après 10 générations la valeur maximale de la fonction objectif reste 0.75. cette valeur est rélativement élognée de la valeur optimale de la fonction de 0.92 obténue par

l'application du backtracking (voir 4.4.1.2.). Pour réduire cette différence entre les deux solutions soit on augmente la population initiale, soit on multiplie le nombre de générations des individus.

## 4.5. Analyse des résultats

Nous avons testé les méthodes d'optimisation présentées sur l'ensemble des données pour le désassemblage des produits avec une structure physique simple. Pour un nombre réduit des composants la méthode exacte qui utilise la technique backtraking fournit l'optimum global de la fonction objectif dans un temps court. Si on augmente le nombre de composants du produit, la complexité de la méthode augmente exponentiellement avec ce nombre, donc le temps de calcul augmente aussi. Par contre, les AG fournissent un optimum local dans un temps convenable et leur complexité augmente avec le nombre de générations. L'avantage des AG est qu'ils fournissent des bonnes solutions en cherchant dans une direction apparemment moins prometteuse. Les résultats donnés par l'algorithme Kangourou sont des optimums locaux et les calculs ont une complexité très réduite.

L'utilisation d'une méthode d'optimisation est au choix du chercheur. Si le produit à désassembler n'a pas une structure très complexe et il ne nécessite pas trop des opérations de désassemblage, mais on a besoin d'une ligne parfaitement équilibrée on choisit d'appliquer une méthode exacte. Si le nombre de composants monte en-dessus de 30, et on a besoin d'un bon équilibrage, les méthodes efficaces pour réaliser l'ordonnancement de la ligne de désassemblage sont les méthodes qui utilisent les algorithmes évolutifs (comme les AG ou les algorithmes stochastiques). Dans les figures 27 et 28 sont représentées la complexité et la convergence des méthodes présentées en fonction de la complexité du produit.

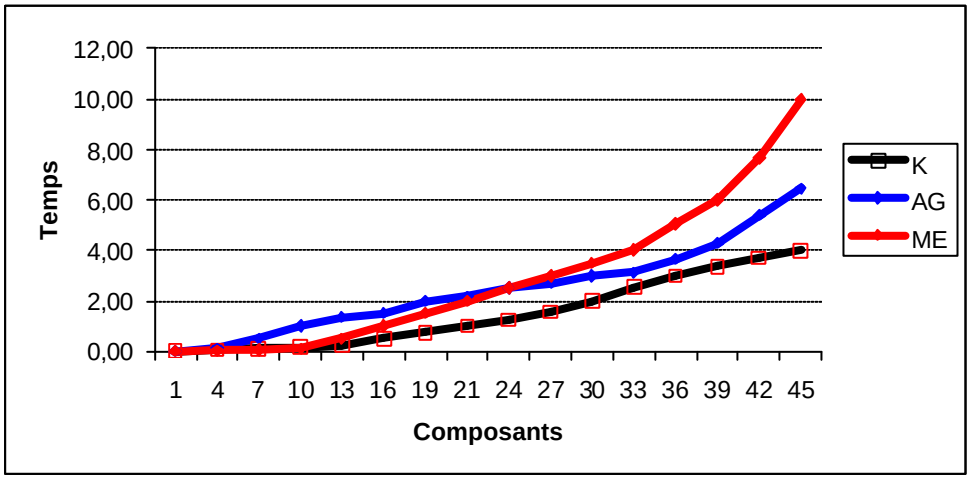


Fig. 27. L'agumentation de la complexité des méthodes proposées

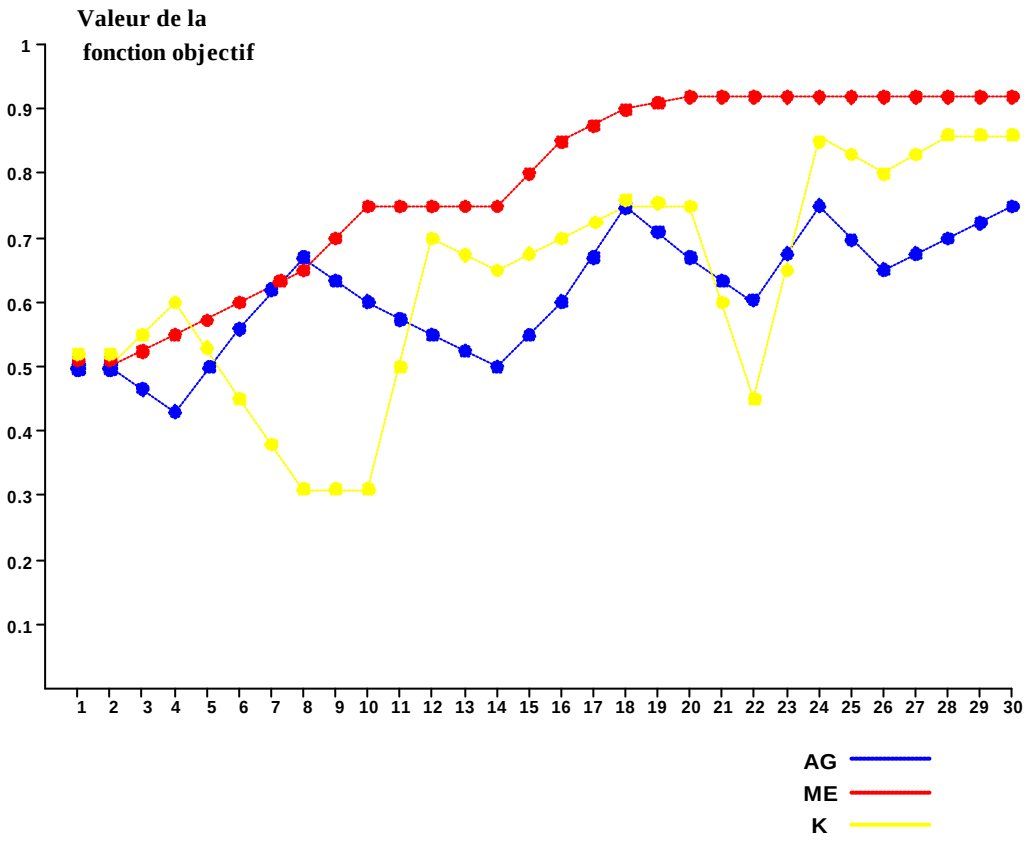



Fig. 28. La convergence des solutions vers l'optimum
(AG - les algorithmes génétiques, K - l'algorithme Kangourou, ME - méthode exacte)

## 4.6. Conclusions

Pour réaliser l'ordonnancement des lignes de désassemblage nous avons présenté dans ce chapitre une méthode classique d'optimisation combinatoire et deux méthodes approchées. L'efficacité de chaque méthode a été évaluée par rapport aux valeurs de la fonction objectif.

Nous avons proposé deux fonctions objectif : la fonction d'équilibrage de la ligne de désassemblage et la fonction de coût (le revenu du désassemblage divisé par le temps de cycle). Nous avons évoqué une troisième fonction : une somme pondérée de l'équilibrage et du revenu. Cette dernière prend bien en compte les deux problèmes clefs du processus de désassemblage : le gain et l'équilibrage, mais malheureusement c'est un problème multicritère où le poids entre les deux critères et difficile d'établir.

La méthode classique nous a donné comme résultat l'affectation optimale pour que le coût soit minimal. Toutefois, nous avons obtenu un bon équilibrage de la ligne de désassemblage. Les méthodes approchées sont utilisées quand on cherche rapidement une valeur optimale de la fonction objectif. Cette valeur représente souvent un optimum

local. L'avantage de la méthode exacte est qu'elle fournit l'optimum global de la fonction objectif, mais le désavantage principal est le développement exponentiel de la complexité de la méthode. Dans l'application des méthodes approchées, la progression vers l'optimum est facilitée par l'apport d'opérations stochastiques. Toutefois, il y a des problèmes dans lesquels les données produisent un effet de seuil indétectable avec un algorithme héuristique qui abandonnerait la recherche dans une direction qui semble peu prometteuse. Dans ce cas les algorithmes génétiques ou les algorithmes stochastiques deviennent avantageux.

La réalisation et l'évaluation de l'ordonnancement en temps réel sont des problèmes qui font appel aux techniques d'aide à la décision. Pour équilibrer la ligne de désassemblage en temps réel il faut connaître à chaque moment les valeurs des coefficients définis dans le paragraphe 3.2.2.2. Suite aux valeurs de ces variables une opération est affectée à un poste ou à l'autre et elle est accomplie dans un mode destructif ou propre. De même, il faut collecter toutes les informations en temps réel et il faut établir les valeurs des variables de commande. Dans le processus de désassemblage du produit il y a des situations quand l'opération de désassemblage d'un composant ou sous-ensemble ne peut pas être accomplie dans un mode propre. Il faut décider s'il est convenable d'appliquer des opérations destructives ou si l'opération est abandonnée pour passer au désassemblage d'un autre composant. Dans les systèmes manuels, le décideur est l'opérateur qui choisi le mode de désassemblage en fonction de son expérience, des outils disponibles et de sa capacité d'évaluer le rapport entre le gain et les pertes après le désassemblage. Dans les systèmes semi automatisés ou automatisés un système informatique d'aide à la décision est nécessaire pour assister le décideur (l'opérateur) à prendre une décision en ce qui concerne la choix du mode de désassemblage, de la gamme optimale, de l'ordonnancement des opérations et éventuellement de l'ordre d'entrée des produits sur la ligne.

# CHAPITRE V

# ARCHITECTURE DU SYSTÈME DE COMMANDE POUR L'ORDONNANCEMENT EN TEMPS RÉEL DU DÉSASSEMBLAGE

## 5.1. L'architecture générale d'un système informatique d'aide à la décision

De nos jeurs, la technologie informationnelle a le pouvoir d'influencer le fonctionnement des sociétés à tous les niveaux : informationnels, organisationnels et industriel. Le mode d'utilisation d'information et la vitesse de décision sont des caractéristiques importantes dans l'évaluation de la performance d'une organisation. C'est pourquoi des systèmes informatiques sont utilisés pour assister l'homme dans les situations critiques de décision.

La décision est le choix d'une direction ou d'une stratégie d'action et elle est prise avec la conscience du décideur [Filip, 2005]. Le décideur est une ou plusieurs personnes autorisées de prendre une décision. Les décisions sont prises dans les situations décisionnelles critiques induites par le changement de l'état du système, par les pannes, les défauts, les perturbations inacceptables, ou par le changement des objectifs de décideur. Le phases du processus décisionnel sont : collecter les données sur l'état du système qui nécessite l'intervention de décideur, analyser et évaluer les alternatives d'action et la prise de décision. Le problème est que la prise de décision a toujours un risque : le risque de ne pouvoir pas accomplie les objectifs du problème. Ainsi, la qualité d'une décision n'est jugé pas seulement par le résultat obtenu, mais par la robustesse de la solution qui minimise le risque.

Les situations décisionnelles pendant le fonctionnement d'un système de production sont de plus en plus complexes. Dans l'ordonnancement par exemple, l'affectation des tâches aux postes de travail est à la fin un problème de prise de décision. Pendant la production une situation critique peut apparaître. Le calcul des affectations possibles dans une telle situation dépasse parfois le temps disponible et la capacité de l'homme. Ainsi, dans une situation d'urgence, un ordinateur devient un instrument de décision très nécessaire. Un algorithme d'ordonnancement implémenté dans un système informatique donne la solution de la situation critique et aide à éviter les interruptions dans le flux de production.

Les systèmes informatiques d'aide à la décision (SIAD) sont des systèmes comple
pour la modélisation et simulation d'une situation décisionne¨ ue d'assis
décideur de prendre la meilleure décision dans ce qui concern fonctionn
du système de production. Avec tels systèmes informatiques on trouve rapidement des solutions aux niveaux de la planification, de l'ordonnancement et de la commande du système de production. De même, ces SIAD sont utilisés dans le cas d'apparition des conflits ou des critères de décisions multiples. La plupart des systèmes informatiques d'aide à la décision font appelle aux techniques d'intelligence artificielle. Souvent, des systèmes experts et des basses de connaissances sont intégrés dans le système d'aide à la décision. Toutefois, ces systèmes sont utilisés pour résoudre des problèmes dans un contexte insuffisamment défini avec des paramètres qualitatifs.

Dans le milieu industriel le système d'aide à la décision doit réagir en temps réel. F. Filip dans [Filip, 1999] exemplifie le fonctionnement de tels systèmes réactifs qui donnent rapidement une bonne solution dans une situation de crise pendant la production (une panne, un stock vide, une erreur de commande). Les décisions dans le milieu industriel doivent être prises sous la pression du temps due à la dynamique du système de production et nécessitent une supervision permanente du processus de production. Dans ce contexte, la réalisation en temps réel d'un système efficace d'aide à la décision est difficile. Pourtant, les nouvelles technologies informationnelles augmentent la vitesse d'enregistrement des informations et fournissent l'environnement informatique nécessaire pour l'implémentation d'un logiciel efficace d'aide à la décision.

Généralement, l'architecture d'un système d'aide à la décision intègre des basses de données, un module de gestion des données et des modèles, un simulateur, une interface avec l'utilisateur et un système de communication entre ces modules.

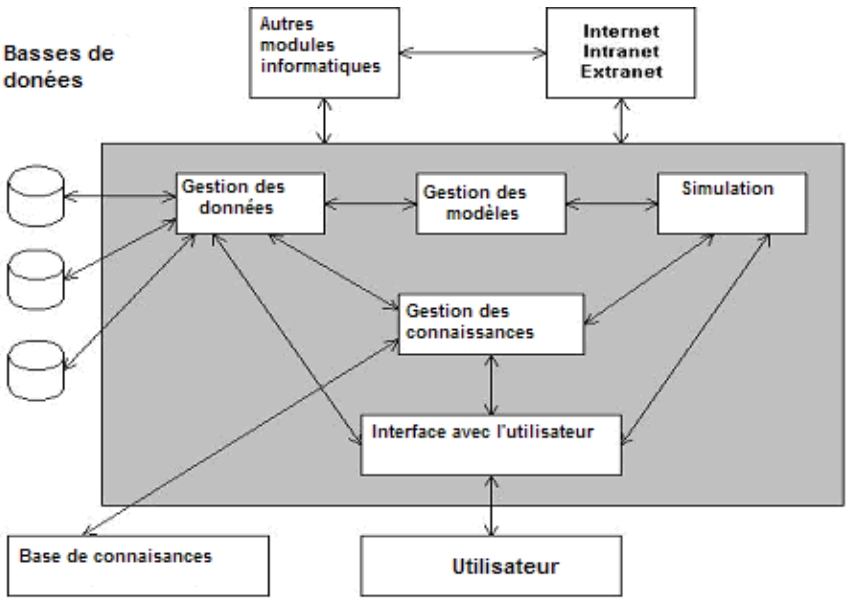



Fig. 28. L'architecture générale d'un SIAD (après [Zaharie, 2001])

## 5.2. Proposition d'architecture SIAD dédié au désassemblage

Les performances d'un système de désassemblage sont évaluées par rapport des critères d'optimisation du processus. Le critère classique est le critère économique. En fonction du rapport entre le coût de désassemblage d'un composant et sa valeur de fin de vie, on prend la décision de le désassembler ou non. Les informations nécessaires dans un processus de désassemblage sont nombreuses : la géométrie de produit à désassembler, les liaisons entre les composants, les matériaux utilisées dans la structure des composants, les valeurs de fin de vie, les temps et les coûts opérationnelles, les gammes de désassemblage, le graphe de précédence, la structure de la ligne, le temps de cycle, etc. Ainsi, les dimensions de la base de données pour un produit à désassembler sont considérables. Une solution est de diviser les informations dans des bases de données dédiées au produit et des bases de données nécessaires au contrôle et à la commande du processus lui même. Les informations sont reliées, c'est à dire que pour un composant manquant ou détérioré, par exemple, le temps et le mode de désassemblage se change, la valeur de fin de vie du produit peut baisser, la ressource utilisée est changée. L'architecture générale du système de gestion des bases de données est représentée dans la figure 29. Néanmoins, la figure n'est pas complète. Parmi les informations nécessaires pour le désassemblage du produit on trouve aussi les gammes de désassemblage, les probabilités de désassemblage propre ou destructif, les temps opératoires, la demande pour chaque composant récupérable. Dans le tableau 5 quelques informations mémorisées dans ces bases de données sont
spécifiées.

Tableau 5 Les bases de données d'application

| ***Base de données du produit à désassembler*** | ***Base de données du processus de désassemblage*** |
|---|---|
| Dimensions géométriques des composants | Temps opérationnels (pour le désassemblage propre ou destructif) |
| Paramètres fonctionnels | Graphe de précédence |
| Matériaux | Séquencement des opérations |
| Liste des composants et sous-ensembles | Ordonnancement des opérations |
| Défauts, déformations, manque de composants | Temps de cycle |
| Probabilités de désassemblage | La demande de chaque composant |
| Options de fin de vie | Ressources |
| Revenus obtenus à la fin de la revalorisation des composants ou du recyclage des matériaux | La structure de la ligne et l'emplacement |
| Les gammes de désassemblage | Type de processus de désassemblage (automatisé ou manuel) |
| Déchets et matériaux polluants | Mode de la collecte des informations du processus (capteurs, dispositives d'arrête, cameras) |

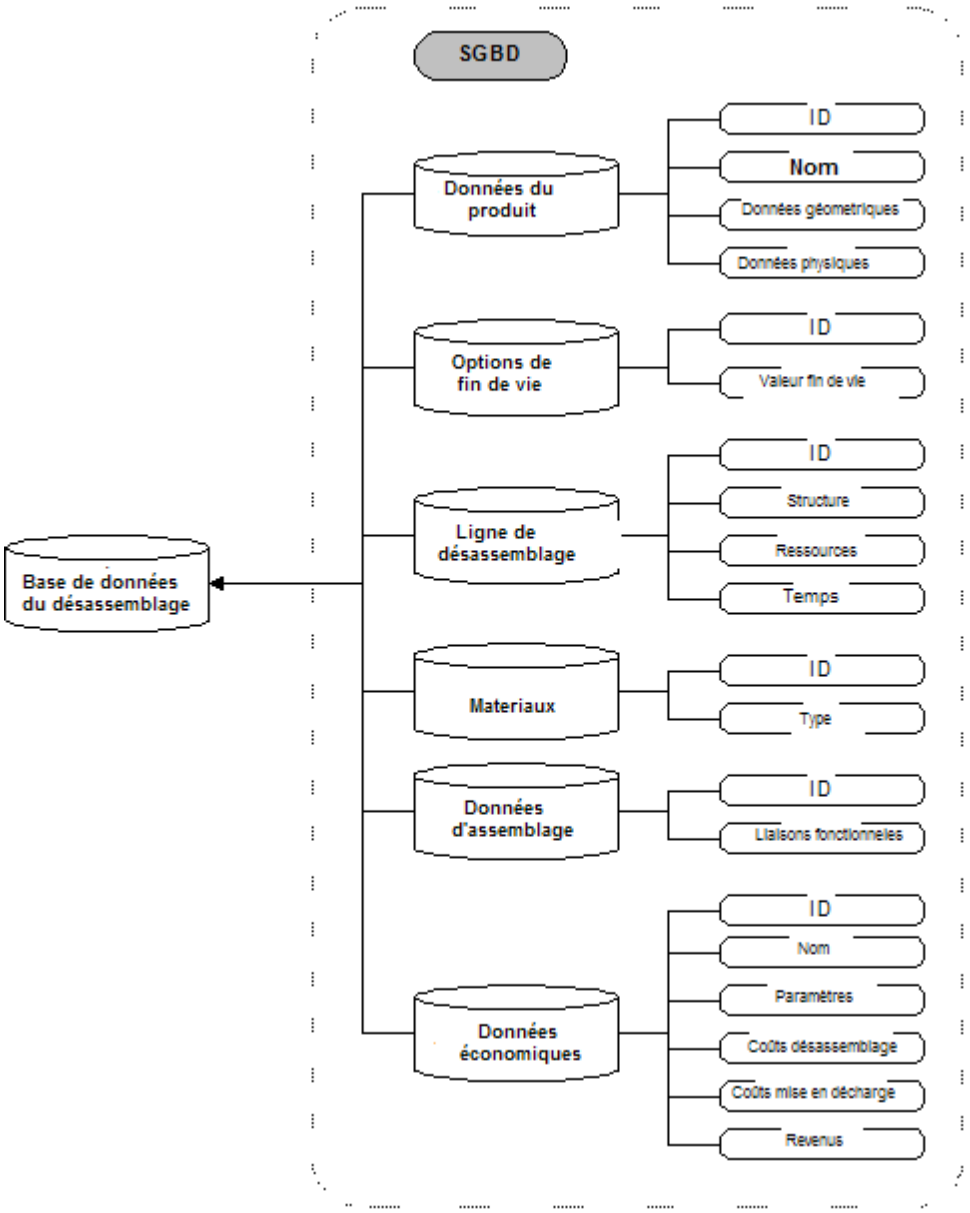



Figure 29. Le système de géstion des bases de donées

Selon l'architecture du système de désassemblage, les éléments principaux du système informatique associé contient un planer et un simulateur. Le planer analyse les données, il fournit le modèle générique du produit et il génère des nouvelles informations nécessaires au système informatique. De même, le planer fournit les gammes possibles de désassemblage. Un système de vision artificielle est nécessaire pour actualiser les informations envoyées au planer et pour ajuster le modèle. Le rôle du simulateur est de mettre en évidence les gammes optimales, les temps opératoires et le séquencement des tâches. Dans ce qui concerne l'architecture des bases de données, un modèle relationnel permet d'établir les actions suivantes pour la séparation du composant ciblé.

La figure 30 représente l'architecture d'un système de désassemblage automatisée, mettant en évidence les parties du système informatique.

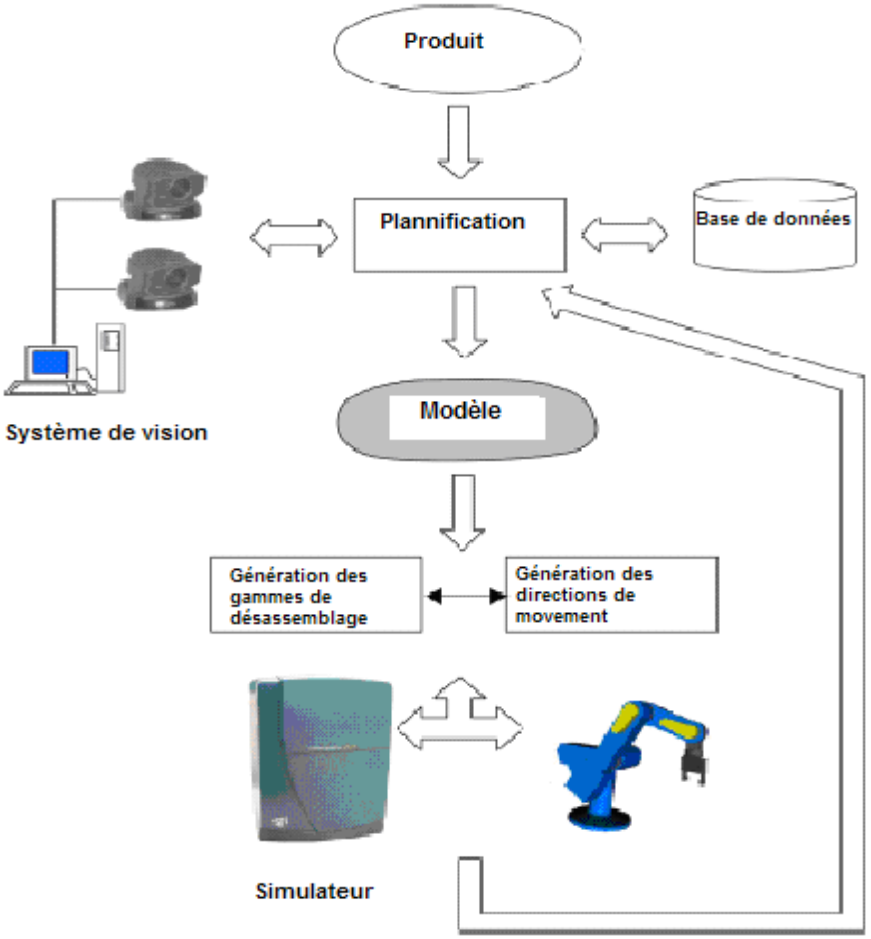



Fig. 30. Les composants d'un système informatique pour le désassemblage

Cette architecture est indiquée pour le désassemblage des produits qui ne subissent pas des modifications essentiales dans la structure des composants pendant leur vie, comme les téléphones ou les ordinateurs portables [Puente, 2002]. Une modification dans la structure anthrène une modification de la gamme de désassemblage. La méthode de représentation des gammes de désassemblage doit permettre telles modifications. Un tableau comparatif des méthodes de représentation est présenté ci-dessous :

Tableau 6 Niveaux de représentation des gammes de désassemblage

| *Niveau de représentation* | *Méthode* | *Avantages* | *Désavantages* |
|---|---|---|---|
| ***Produit*** | Graphes de liaisons | Représentation intuitive Présentation de toutes les liaisons entre les composants | Il faut simplifier le graphe avant de calculer les gammes optimales |
| ***Elaboration des gammes*** | Graphes de précédences | Le calcul rapide de la gamme optimale par rapport au coût de désassemblage | Il faut connaître les connections entre les composants |
| | Graphes AND/OR | Les types des liaisons entre les composants sont connus | Nécessitent des représentations itératives pour trouver la gamme optimale |
| | Arbres de désassemblage | Possibilité d'utilisation en temps réel Augmentent la vitesse de calcul de la gamme optimale | Nécessite d'espace pour la représentation de toutes les relations entre les composants |
| | Réseaux de Petri | Indiquées pour l'analyse en temps réel | La complexité du réseau augmente avec la complexité du produit |
| ***Optimisation*** | Réseaux neuronaux | Représentation de la fonction de coût | Difficulté en représentation des relations logiques |
| | Système expert | La base de connaissance est utilisée pour le désassemblage des produits similaires ou de la même famille | Nécessite beaucoup des informations sur le produit |
| | Algorithmes génétiques | La gamme optimale est obtenue rapidement | Le codage difficile et la solution qui n'est pas toujours optimale |

Pour la représentation des gammes nous avons utilisé les Réseaux de Petri de désassemblage. Sur tel réseau on peut marquer les temps opératoires, l'état courant du produit, les opérations alternatives destructives, les revenus de fin de vie [Zussman, 1999].

Le modèle générique du produit à désassembler est généré par le planer qui analyse l'ensemble des données d'entrer du système de vision artificiel et les signaux des capteurs avec les informations contenus dans les bases de données ou dans les bases de connaissances. De même il génère les gammes de désassemblage éventuellement en

mettant en évidence les valeurs de fin de vie. C'est le tour du système d'aide à la décision de trouver la meilleure gamme selon les critères d'optimisation prises en considération et d'établir l'affectation des opérations aux postes de travail. Si le système reçoit en permanence des informations sur le développement du processus, la prise de décision est fait en temps réel. Pour les produits qui change la structure géométrique des composants pendant leur utilisation, le planer est remplacé d'un système virtuel de désassemblage où les opérations sont accomplis au début par la personne spécialisé et puis par le robot qui a appris les mouvements [Garbaya, 2003].

La méthode d'ordonnancement en temps réel nécessite un système interactif d'aide à la décision. Les informations sont collectées en temps réel (événements, données, paramètres, décisions) et la commande est effectué à la suit d'analyse des ces informations. Les procédures d'aide à la décision ont pour objectif l'assistance du décideur dans le processus de prise de décisions.

Le système SIAD mémorise l'état courrant du processus de désassemblage soit sous la forme du réseau de Petri soit dans des fichiers de données, ce que lui permet à chaque instant de connaître l'ensemble de décisions possibles à l'apparition des évènements. Il mémorise également l'ensemble d'ordonnancements prévisionnels sous la forme du graphe de précédences ce qui lui permet connaître l'admissibilité de l'ordonnancement courant.

Le SVA[4] mise à jour les informations sur l'état du produit. Des messages et des requêtes sont envoyées par les ressources vers le système d'aide à la décision dans un mécanisme de communication utilisant des files d'attente. Pour chaque requête le système identifie les entités et les opérations concernées et renvoie des informations sur ces entités. Si le message est une déclaration de décision, le système vérifie l'état courrant du processus et si la décision est valide il envoie un message de test positif au système de commande qui change l'état courrant. En général, la réponse à une décision est constituée d'un nouvel état de la ressource concernée, de la liste des opérations possibles ou des leurs affectation. Le graphe de potentiels tâches est mise à jour et les opérations sont actualisées en fonction de la date courante. Le nouvel état génère des nouvelles informations qui sont renvoyées vers le SIAD. Si le test est négatif, c'est à dire que l'état courrant du processus ne permet pas l'application de la décision, une information de requête non valide est envoyée au système de commande qui envoie à son tour une requête d'informations supplémentaires. Dans le cas d'apparition d'un événement imprévu (tel une panne de ressource, ou l'impossibilité de continuer le processus), le système d'aide à la décision envoie un message d'erreur vers le système de commande qui peut changer le mode d'exécution d'opération ou le flot de production. La commande est réactive et est donnée à la suite d'une comparaison des informations reçues du système et les valeurs prévisionnelles.

[4] Système de vision artificielle

L'architecture générale d'un système informatique de control du désassemblage est proposée dans la figure suivante :

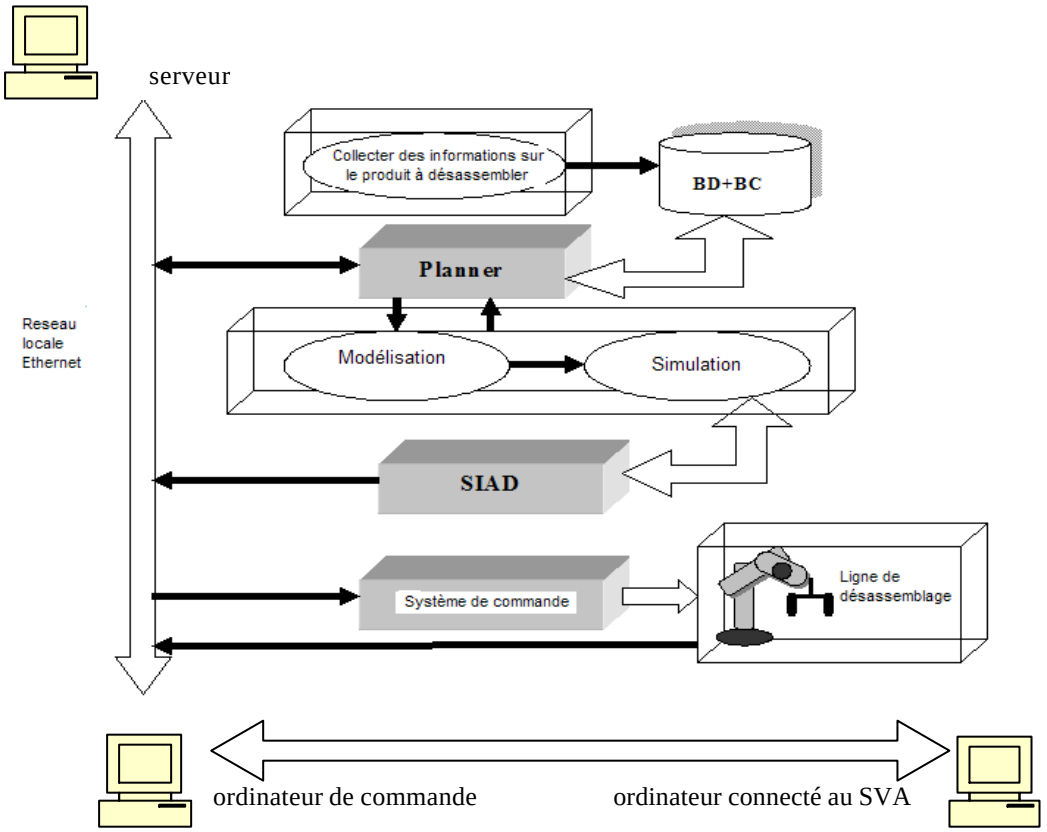



Fig. 31. Architécture du système informatique de control du désassemblage

Le SIAD est souvent compris dans l'architecture du système de commande. Entre les deux modules il y a un permanent changement d'informations en vue de trouver la meilleure solution dans une situation donnée.

## 5.3. Implementation de l'architecture proposée

### 5.3.1. Arbres de décision : instruments graphiques d'aide à la décision

Les instruments graphiques ont toujours été les plus utilisées du à leur capacité visuelle de représentation. Dans le domaine de l'analyse à la décision, les arbres de décision sont peut-être les instruments visuels les plus utilisés pour représenter les alternatifs de décision. Ils sont faciles à représenter et analyser, ils offrent un instrument de calcul efficace et leur modélisation et simulation est spectaculaire.

5.3.1.1. Notions théorétiques

Un arbre de décision est un diagramme qui représente une séquence logique entre les événements de chance et de décision dans un processus [Targett, 1996]. Les symboles utilisés dans ce diagramme sont :

□ Point (événement) de décision

○ Point d'apparition d'événement type chance

— La liaison logique entre les événements

L'exemple suivant illustre très bien ce concept :

Supposons que nous devons nous décider si nous acceptons un pari. La décision est difficile donc on choisi d'utiliser une méthode heuristique pour nous aider dans la prise de décision. En faite, nous allons utiliser une monnaie et le choix aléatoire d'une facette va nous dire d'accepter ou pas le parie. Théoriquement il y a 50% chances que la monnaie tombe sur une facette ou l'autre. Si la monnaie tombe sur la facette choisie, nous allons gagner 5 unités de valeur, si non nous allons perdre 4 unités. L'arbre de décision correspondant est représenté dans la figure 31.

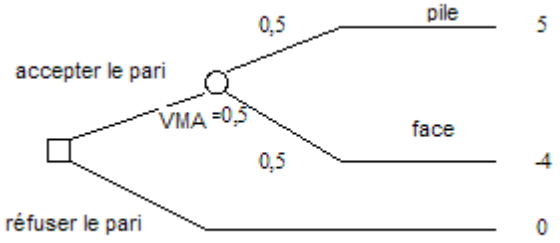



Fig. 32. Arbre de décision (aprés [Targett, 1996])

A la fin de chaque branche de l'arbre il y a le gain ou le perte du décideur s'il choisit cette branche (les pertes sont négatives). Les noeuds évenement sont soumises au hasard, donc à un évènement imprévu. On associe des probabilités aux branches qui sortent d'un noeud évènement. Pour évaluer les alternatives de décision, une valeur monétaire est calculée : la valeur monétaire attendue VMA. C'est l'espèrance mathématique associé à l'évènement considéré. Dans la théorie d'analyse décisionnelle cette valeur est utilisée comme critère économique. Dans l'exemple précèdent sa formule de calcul est :

$$VMA = (0,5 \times 5) - (0,5 \times 4) = 0,5 \qquad (27)$$

Le profit le plus grand est obtenu à l'acceptance du pari (VMA=0,5>0) alors le décideur va accepter le pari et assumer le risque.

5.3.1.2. La transformation du réseau de Petri de désassemblage en un arbre de décision du désassemblage (ADD)

Le but de cette opération est de fournir au décideur (l'homme) un instrument d'aide à la décision en vue de trouver la gamme optimale qui maximise le revenu final *pendant* le processus de désassemblage.

**Données :**

- le réseau de Petri de désassemblage (les gammes de désassemblage)
- les temps opératoires
- les options de fin de vie et les revenus correspondants pour chaque composant ou sous ensemble
- les probabilités de désassemblage (propre ou destructif)

**Hypothèses :**

- la probabilité de désassembler un composant dans une manière propre est connue par l'expérience
- il y a un seul type de produit à désassembler
- les temps opératoires sont connus et déterministes
- les coûts opératoires sont pris en considération dans une étape antérieure au calcul des revenus finaux
- l'impossibilité et l'abandon d'opération de désassemblage due aux déformations des composants sont prises en considération

**Notations :**

**P** l'ensemble de *n* places du réseau de Petri
**T** l'ensemble de *m* transitions du réseau de Petri
**P'** l'ensemble de *q* places finales du réseau de Petri
$\mathbf{R_i}=\left(R_i^+\right)$ ou $\mathbf{R_i}=\left(R_i^-\right)$ l'élément *i* du vecteur de revenus calculé au moment courant de désassemblage; le gain est un revenu positif, la perte est répresentée par un nombre négatif $\mathbf{A_{mxn}}$ la matrice d'incidence pour le réseau du Petri avec les éléments :

$$a_{ij}=\begin{cases} -1 & \text{si } T_i \text{ est une transition qui suite la place } P_j \\ 1 & \text{si } T_i \text{ est une transition avant la place } P_j \\ 0 & \text{autrement} \end{cases} \quad \text{où} \quad i=\overline{1..m} \quad j=\overline{1..n}$$

$\mathbf{C_{mx2}}$ la matrice des probabilités données par le rapport du succès des opérations de désassemblage avec les éléments :

$c_i=\left(p_i^+, p_i^-\right)$ où $p_i^+$ *la probabilitéderéussitedel'opérationopide désassemblage* et $p_i^-$ *la probabilitéd'échecdel'opérationopi dedésassemblage* $i=\overline{1..m}$

Il faut tenir compte que $p_i^+ + p_i^- = 1$;

L'algorithme de transformation du réseau de Petri dans un arbre de désassemblage a les étapes suivantes :

**Début de l'algorithme ADD**

**Etape 1.** *La construction de l'arbre de décision :*

- A chaque place non finale $P_j \notin P^{'}$ un noeud de décision est associé $D_j$ (où *j =1 ..n-q* ) ;
- A chaque transition $T_i \in T$ on noeud dévènement est associé $C_i$ (où *i = 1 .. m* ) ;
- Les alternatives dans un noeud de décision sont : de continuer le désassemblage suivant la gamme prévue, ou d'abandonner l'opération si elle devient trop destructive ou si elle est considérée impossible;
- Chaque opération a deux possibilités : de réussite ou d'échec avec une certaine probabilité
- Le calcul des éléments de la matrice d'incidence de l'arbre de décision $\mathbf{B_{mx(n-q)}}$ :

$$b_{ij} = \begin{cases} 1 & \text{si} \quad a_{ij} = 1 \\ -1 & \text{si} \quad a_{ij} = -1 \\ 0 & \text{si} \quad a_{ij} = 0 \end{cases} \text{ où } \begin{matrix} i = 1..m \\ j = 1..(n-q) \end{matrix}$$

**Etape 2.** *Le calcul du revenu final*

A chaque noeud final *j* de l'arbre de décision après la formule

$$R_j = \sum_k r_k \qquad (28)$$

où les $r_k$ sont donnés et représentent les revenus associés à chaque composant ou sous ensemble du produit

**Etape 3.** *Le calcul des probabilités*

A chaque noeud évènement $C_i$ deux probabilités sont associées $p_i^+$ et $p_i^-$ ; les probabilités sont données dans la matrice **C**;

**Etape 4.** *Le calcul de la valeur monetaire*

Le calcul de la VMA à chaque noeud évènement après la formule

$$VMA(C_i) = c_{ij} \cdot R_i = R_i^+ \cdot p_i^+ + R_i^- \cdot p_i^- \qquad (29)$$

**Etape 5.** *Le parcours de l'arbre*

Appeler la procédure de parcours en arrière de l'arbre [Targett, 1996] qui calcule le VMA à chaque noeud de décision en commencer avec les noeuds finaux et finir avec la racine de l'arbre.

**Etape 6.** *L'évaluation des alternatifs*

La meilleure décision prise dans chaque noeud de décision est celle qui donne la valeur maximale de la VMA pour ses alternatives :

$$\max_i \{VMA(C_i), VMA(abandonne)\} \qquad (30)$$

**Etape 7.** *L'evaluation du critère d'arrêt*

Si le noeud courant n'est pas $D_1$ alors répéter l'étape 4.

Sinon **STOP.**

### 5.3.1.3. Exemple d'application de l'algorithme ADD

Nous reprenons un exemple de désassemblage d'un produit simple comme celui du paragraphe 4.3.2. Le réseau de Petri de désassemblage est donné dans la figure 21. Les notations et les données d'entrée sont représentées sur la même figure.

A la suite de l'application de l'algorithme précèdent, l'arbre de décision résultant est présenté dans la figure 32. Le revenus dans les noeuds finaux sont calculés après la formule (28). Les valeurs monétaires aux noeuds de chances sont calculées après la formule (29)

$$R_1 = r_7 + r_6 + r_5 + r_8 = 2{,}5 - 1{,}5 - 3{,}5 + 4 = 1{,}5$$
$$R_2 = r_3 + r_5 + r_8 = -3 - 3{,}5 + 4 = -2{,}5$$

---

$$R_{11} = -7$$

$$VMA(C_4) = 1{,}5 \cdot 0{,}5 - 2{,}5 \cdot 0{,}5 = -0{,}5$$
$$VMA(C_2) = -0{,}5 \cdot 0{,}6 - 2 \cdot 0{,}4 = -1{,}1$$
$$VMA(C_5) = 1{,}5 \cdot 0{,}5 + 1 \cdot 1{,}5 = 1{,}25$$
$$VMA(C_3) = 1{,}25 \cdot 0{,}5 - 2 \cdot 0{,}5 = -0{,}375$$
$$VMA(C_1) = -0{,}375 \cdot 0{,}8 - 7 \cdot 0{,}2 = -1{,}7$$

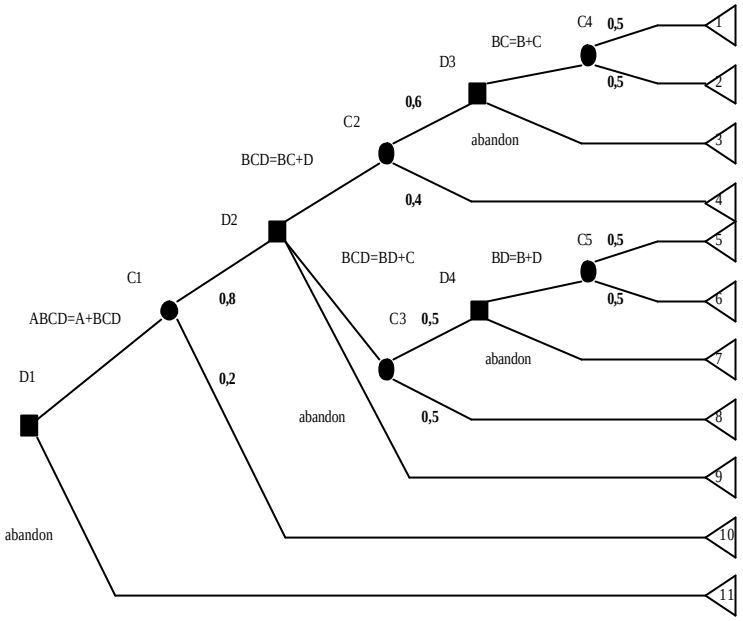



Fig. 33. Arbre de décision pour le récépteur téléphonique

On applique la procédure de parcours de l'arbre et à chaque pas la valeur maximale du VMA donne la meilleure décision. La figure 33 represente l'arbre de décision après.

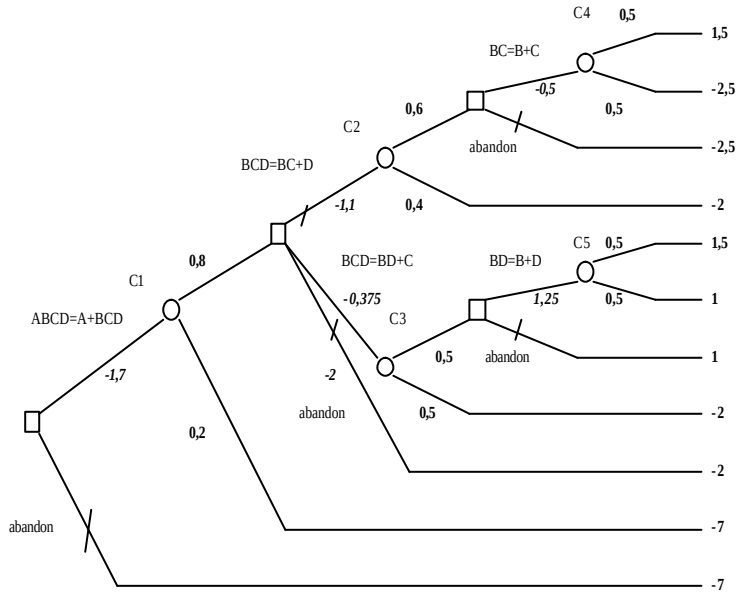



Fig. 34. Arbre de décision final

Les branches avec des alternatives de décision inefficaces sont coupées. La décision optimale est de séparer A de l'ensemble BCD, puis C de BD, puis D de B.

La conclusion est que la gamme optimale est obtenue après le franchissement (en ordre) de transitions $t_1 \rightarrow t_3 \rightarrow t_5$ . C'est à dire que la séquence opératoire qui maximise le profit est celle que exprime le désassemblage total en vue d'obtenir les quatre composants A,B,C,D. Si, par exemple, le composant B ne peut pas être séparé du composant D (c'est à dire que le circuit imprimé est détruit ou moulé) la décision optimale par rapport à la valeur maximale du profit est celle qui donne un VMA le plus proche de la valeur maximale antérieure. Alors, si les composants B et D sont difficiles à séparer, on essaye de séparer B de C.

5.3.1.3. Les avantages des arbres de décision dédiés au désassemblage

L'utilisation des arbres de décision dans la modélisation des gammes de désassemblage permet l'application des techniques spécifiques de l'analyse décisionelle. Ces techniques incluent l'analyse du risque assumé, l'évaluation de la validité du critère économique, la simulation type Monte Carlo, l'étude de la distribution des probabilités d'apparition d'un événement imprévu, programmation sous incertitude, et ainsi de suite. Ces techniques sont des méthodes simples utilisées pour mettre en ordre les informations et pour évaluer l'influence de la variabilité des données sur le résultat attendu.

Par l'utilisation des arbres de décision en désassemblage, le décideur peut évaluer l'opportunité d'une opération de désassemblage par rapport au critère économique pris en considération (temps, coût, revenu). Cette évaluation peut être faite *en temps réel* grace à la propriété des arbres de décisions de mettre en évidence la meilleure décision à un moment donné. De même, le décideur peut évaluer le risque assumé à la prise d'une décision et la robustesse de sa décision au changement des données d'entrée. Souvent les états des certains composants ne sont pas connus a priori, ainsi les données changent pendant le désassemblage du produit. Les systèmes informatiques d'aide à la décision ont la capacité de lire les nouvelles données, d'actualiser les informations et de les réinterpréter pour recalculer la meilleure décision en ce qui concerne la continuation et le type du désassemblage.

La représentation graphique des arbres de décision est simple et intuitive. La décision optimale est donnée rapidement à la demande du décideur. Il y a des nombreux logiciels qui utilisent déjà l'arbre de décision comme un instrument graphique visuel d'assistance. Un de ces logiciels sera présenté dans le paragraphe suivant.

Ces avantages nous ont déterminé à inclure dans l'architecture du système de commande du désassemblage un système informatique d'aide à la décision.

### 5.3.2. L'interface avec l'opérateur

Le système de commande a pour rôle la comparaison des données stockées dans les bases de données avec les informations qui arrivent du processus. A la suite de cette analyse le système fournit une commande réactive pour minimiser la différence entre les paramètres prévisionnels et réels du processus. Plusieurs interfaces sont nécessaires pour accomplir ces tâches.

Les interfaces ont été implémentées sous Visual C++ 6.0. et elles permettent la manipulation des informations des bases de données conçues sous la forme : ajouter, effacer, modifier les enregistrements. Si le spécialiste en désassemblage donne les gammes de désassemblage, les temps opératoires, les revenus de fin de vie, et autres informations dans un réseau de Petri de désassemblage, l'interface permet la transformation automatique du réseau dans un arbre de décision. L'analyse de décision est faite aussi par l'interface de commande.

Pour la compréhension des éléments de l' interface, nous allons continuer d'illustrer l'architecture du système informatique proposée par l'utilisation de l'exemple présenté dans 4.3.2. : Le désassemblage d'un récepteur téléphonique.

5.3.2.1. Les bases de données

Nous allons diviser les informations en quatre parties. Dans la première base il y a les données sur le produit (le nom, la famille d'appartenance, le domaine d'utilisation, l'état, la probabilité de désassemblage propre, le valeur de fin de vie, le nombre de composants, la fonctionnalité, les dimensions géométriques, les gammes de désassemblage, les liaisons géométriques entre les composants). Nous avons utilisé une représentation relationnelle entre les tables de cette base de donnée. A l'actualisation des tables primaires, tous les enregistrements des tables secondaires sont changés. Les relations entre les tables sont représentées dans la figure 34.

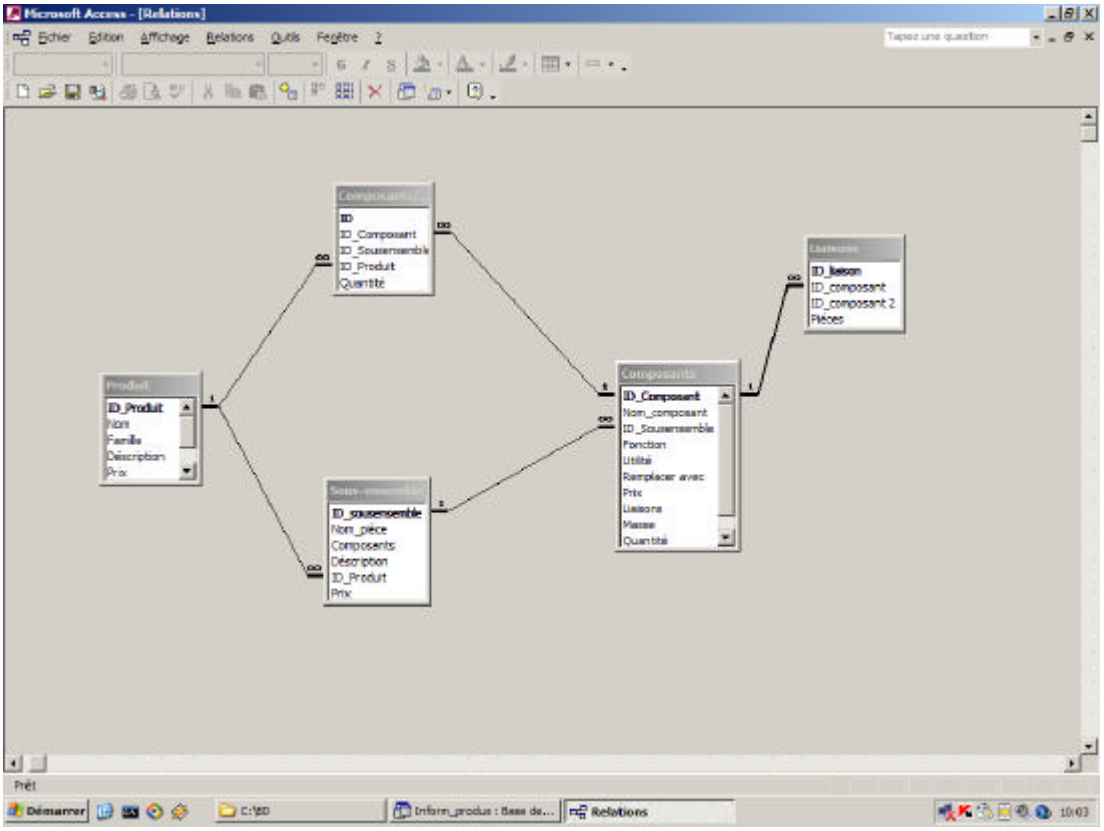



Fig 35. Les relations dans la base de données du produit

Une autre base de données est celle qui contient des informations sur les sous ensembles et les composants du produit : le nom, le nombre de composants du même type, le matériel de fabrication, les options de recyclage ou de revalorisation, éventuellement les coûts de broyage, les coûts de mise en décharge, les coûts de revalorisation, les caractéristiques géométriques, le prix, la masse, le volume, l'état physique et l'état fonctionnel du composant, les opérations et les outils nécessaires pour le désassemblage de chaque composant, le type de liaison entre les composants voisins.

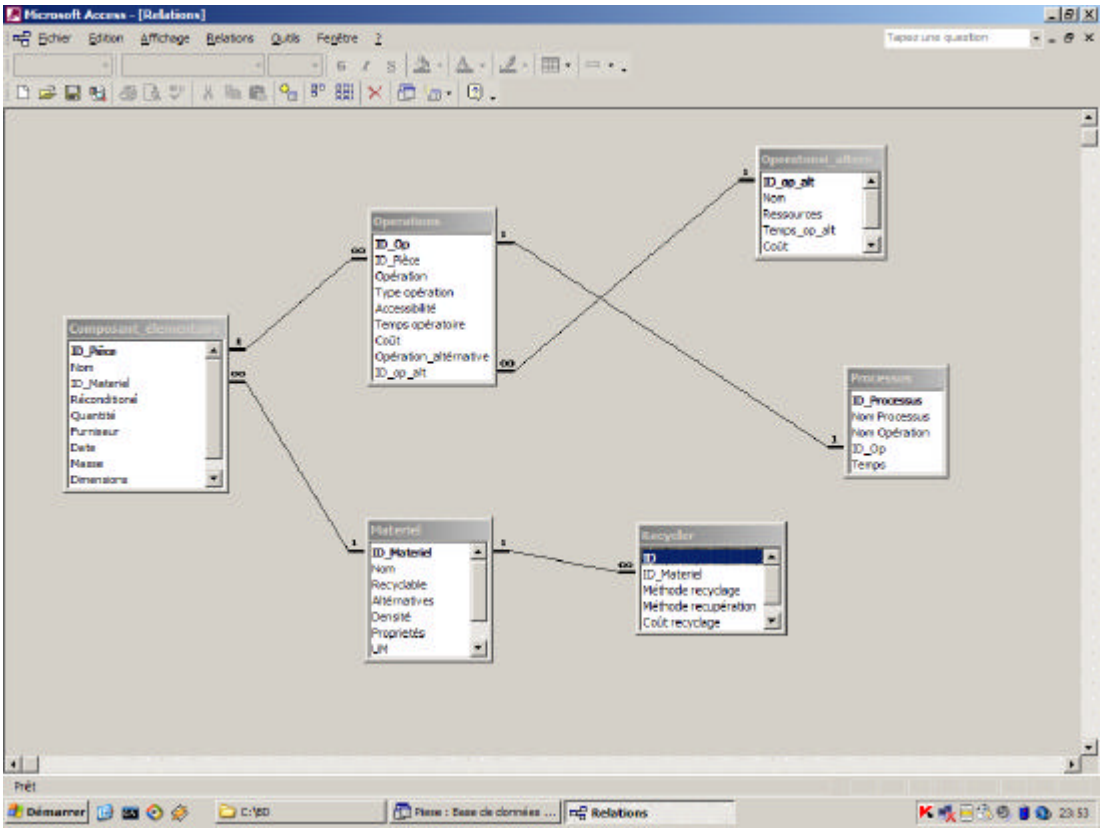



Fig 36 Les relations dans la base de données du composants

On a besoin aussi des informations sur le développement du processus de désassemblage lui même. Il faut connaître les ressources utilisées, la configuration de la ligne de désassemblage, le nombre de postes de travail, le nombre d'opérations, le temps de cycle, les temps opérationnels de désassemblage, les coûts de désassemblage, l'affectation possible des opérations sur les postes, le séquencement du produits sur la ligne, le type de désassemblage de chaque composant (propre ou destructif) et autres informations relatives au processus de désassemblage. Dans la figure suivante, l'interface avec la base de donné du processus est représentée.

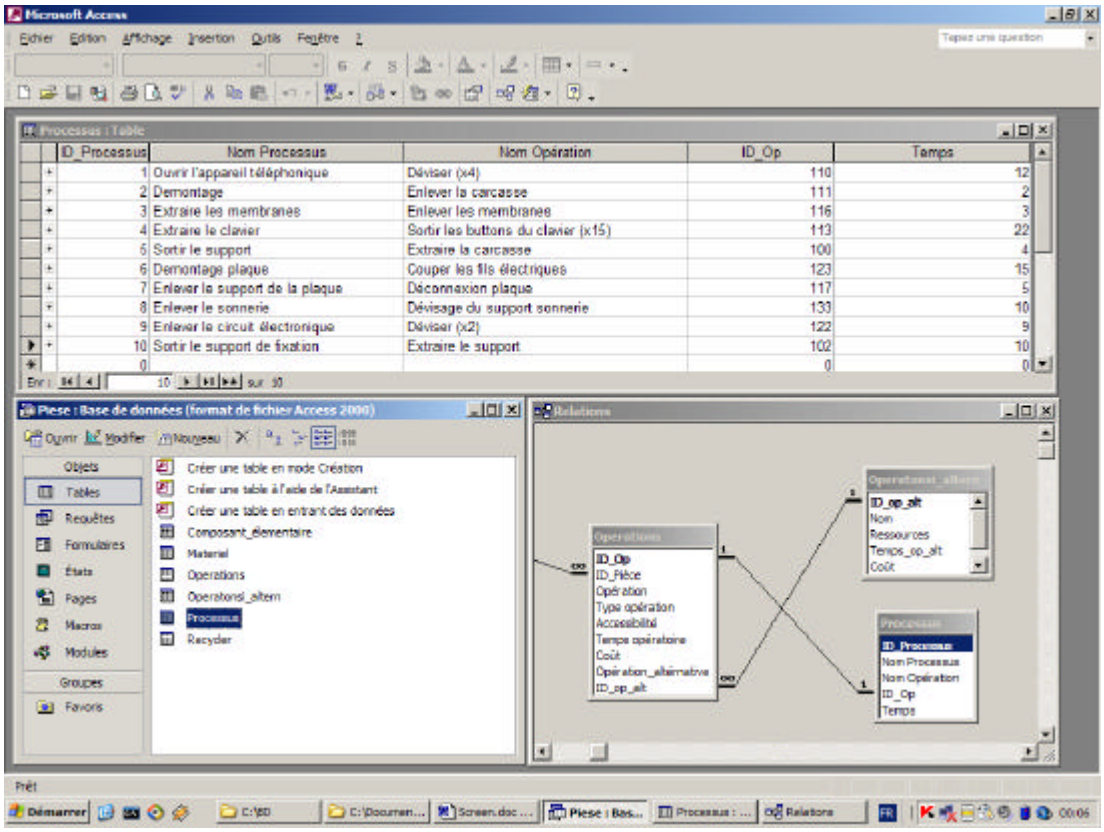



Fig 37. La base de données du processus de désassemblage

Les opérations sur les enregistrements des bases de données sont faies soit par une interface opérateur (Fig. 37), soit directement par l'interface d'application Access.

Les dimensions des bases de données augmentant avec la complexité du produit désassemblé. Si le produit est un produit manufacturier avec une structure complexe (comme une voiture par exemple) l'analyse du désassemblage est fait par ses sous ensembles (le panneau de bord, la porte, l'ensemble boîte de vitesse - moteur). Dans ce cas à chaque sous-ensemble correspond quatre bases de données comme il a été spécifié auparavant.

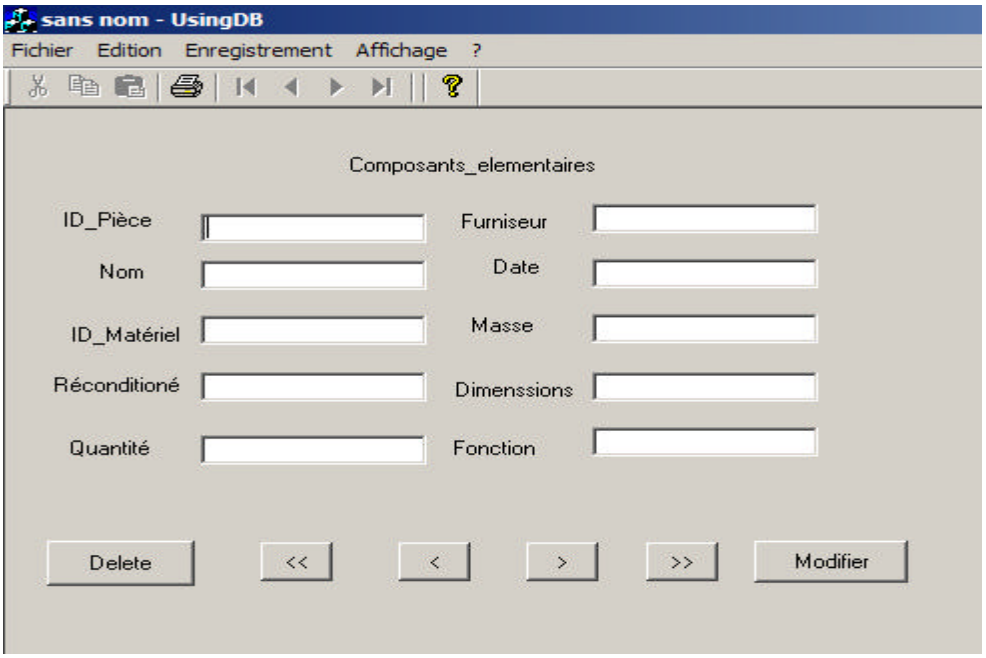



Fig 38. L'intérface opérateur pour modifier les enregistrements

5.3.2.2. L'interface de l'aide à la décision

La première opération est de lancer l'interface qui permet l'introduction des gammes de désassemblage. Il faut préciser qu'on considère des produits d'une complexité réduite (avec 3-4 gammes de désassemblage et 10-15 opérations de désassemblage) en vue de minimiser le temps d'introduction des données et le temps de calcul. pour les produits plus complexes, tels qu'une voitures par exemple, l'application visuelle est utilisée pour chaque sous-ensemble du produit.

Dans la figure 38 est présentée l'interface pour introduire les gammes de désassemblage. L'image du réseau de Petri est importée d'un fichier .BMP. Cette image donne à l'utilisateur la possibilité de visualiser les gammes de désassemblage. Le réseau de Petri est mémorisé dans la matrice de désassemblage. Les revenus ou les pertes dues au désassemblage sont statistiquement anticipées et ils sont introduits dans les champs prévus. Les probabilités de réussite ou d'échec d'un désassemblage propre sont également introduites. Toutes les données sont enregistrées automatiquement dans un fichier texte. L'interface donne la possibilité de visualiser et modifier les enregistrements des bases de données à l'aide du menu "Base de données". De même, par le bouton "Décisions" on fait la liaison avec l'interface d'aide à la décision. Le bouton "Initialiser" efface les données du fichier texte dans le cas d'un changement du produit. Le reseau de Petri est mémorisé dans la matrice de désassemblage. Les revenus ou les pertes dues au désassemblage sont statistiquement anticipés et ils sont introduits dans les champs prévus. Toutefois, les probabilités de réussite ou d'échec d'un désassemblage propre sont introduites. Toutes les données sont enregistrées automatiquement dans un fichier texte. L'interface donne la possibilité de visualiser et modifier les enregistrements des bases de données à l'aide du menu "Base de données".

De même, par le bouton "Décisions" on fait la liaison avec l'interface d'aide à la décision. Le bouton "Initialiser" efface les données du fichier texte dans le cas du changement du produit.

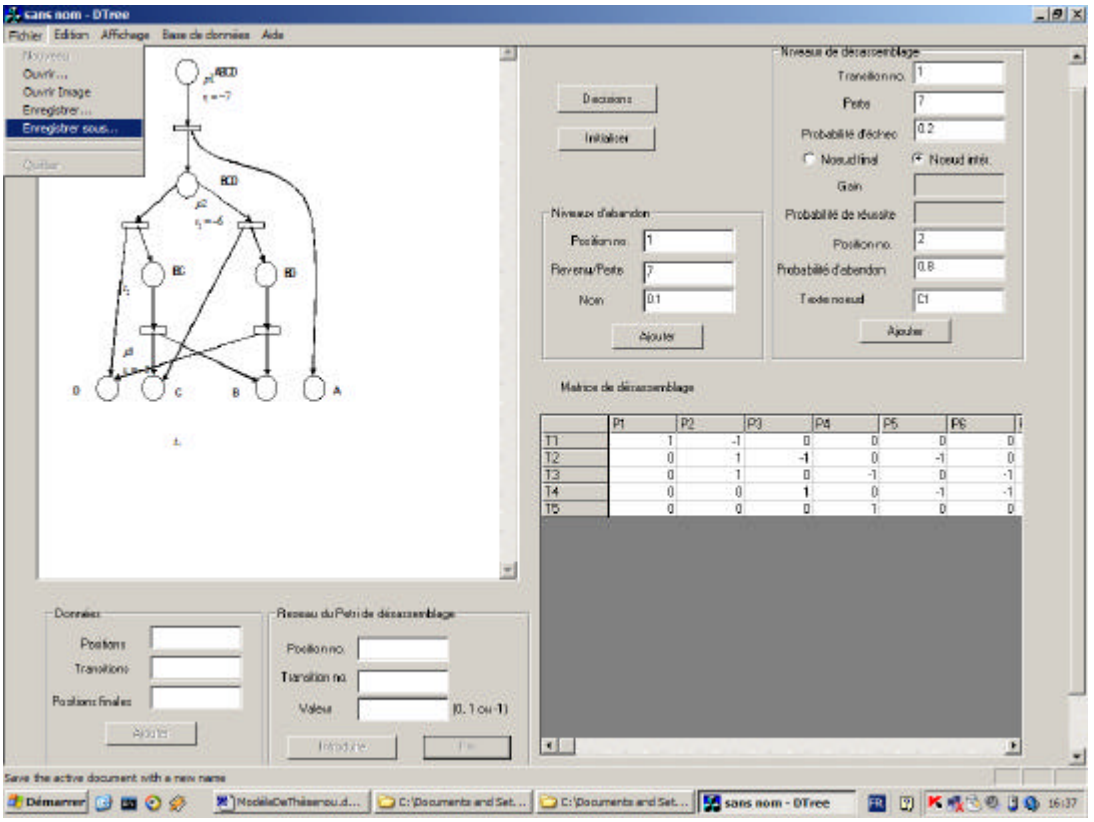



Fig. 39. Le reseau de Petri de désassemblage donnée par sa matrice d'incidènce

L'interface pour la transformation du reseau de Petri dans un arbre de désassemblage, est donnée dans la figure 39.

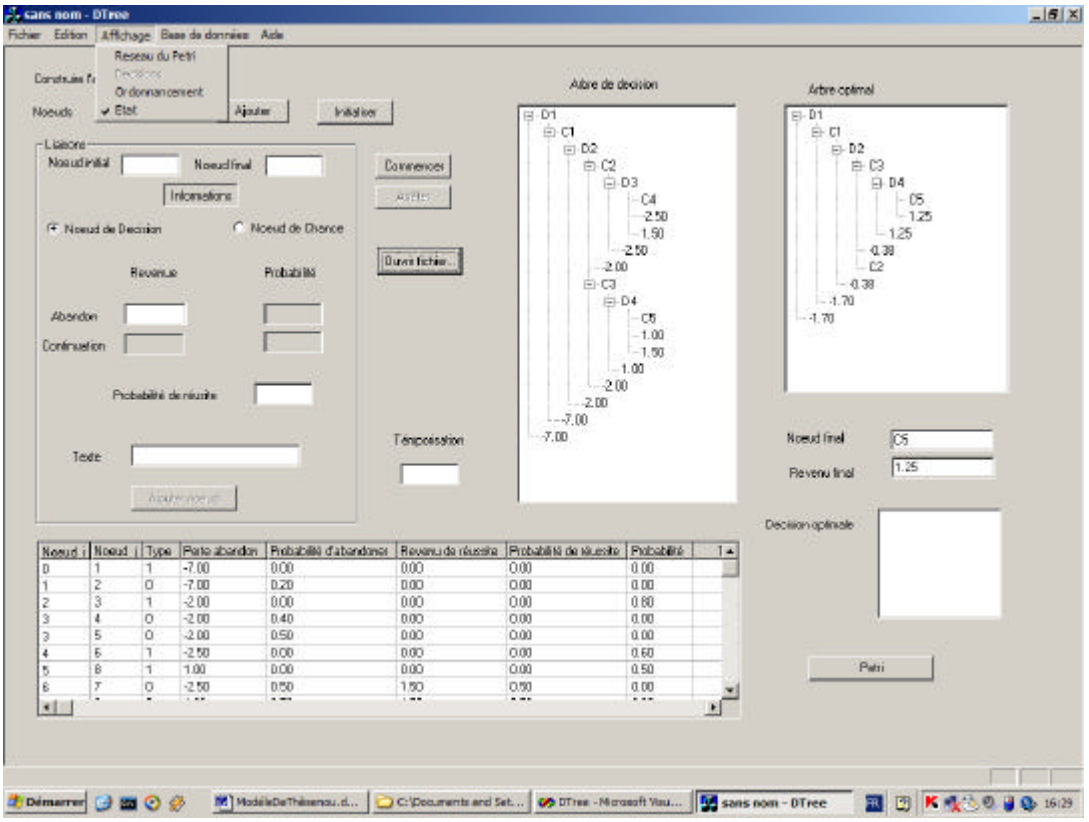



Fig. 40. L'interface d'aide à la décision

L'algorithme de transformation du réseau de Petri [Duta, 2003b] génère un arbre de décision qui permet de visualiser les gammes de désassemblage mais du point de vue économique. Ce qui nous intéresse est le revenu final obtenu à la suite du désassemblage du produit. Le problème qui se pose est comment évaluer ce revenu dans le cas du désassemblage destructif ou pire, dans le cas de l'échec des opérations de désassemblage. Dans ces situations l'analyse décisionnelle est nécessaire. L'algorithme ADD donne la décision optimale. Dans la figure 39 la décision optimale correspond au cas du désassemblage propre total. L'interface de l'aide à la décision donne la possibilité de construire l'arbre de décision par l'introduction des informations concernant les noeuds dans les champs correspondantes. Cette option est utile quand l'utilisateur veut faire une prédiction du développement des opérations. Si non, un ficher texte est lu toutes les 10 secondes pour actualiser les noeuds d'arbre. Si une opération de désassemblage n'est pas possible (si elle est abandonnée) ou si l'opération devient destructive, le fichier d'entrée et la structure de l'arbre de décision évoluent en conséquence. L'utilisateur est celui qui fait le choix : prendre ou non la meilleur décision indiquée par l'assistant informatique.

Le calcul de l'arbre optimal et l'affichage de la meilleure décision a été inspiré par le programme informatique DPL 6.0 (*Décision Programming Language*) présenté dans [Duta, 2003b]. Ce logiciel d'aide à la décision (dans la version avec licence) permet une représentation graphique d'exception des arbres de décision. De même il assure une analyse décisionnelle complète : l'analyse du risque et l'analyse de sensibilité de la solution aux variation des paramètres d'entrée. Par contre, les arbres de décision sont

construits souvent manuellement ou automatiquement à l'aide des diagrammes d'influence. Ceux qui utilisent ce logiciel doivent connaître la théorie de la décision et les instruments graphiques de représentation. C'est la raison pour laquelle DPL est utilisé plutôt par les managers et économistes que par les ingénieurs. L'auteur de ce travail a étudié le mode de fonctionnement du DPL et a adapté ses facilités au problème de désassemblage. De plus, l'interface présentée précdedemment permet la transformation du réseau de Petri en un arbre de décision, l'avantage de cette représentation étant que sur un arbre de décision on peut appliquer les théories d'analyse décisionnelle.

5.3.2.3. L'interface dédiée à l'ordonnancement

Une fois la gamme optimale choisie, le programme permet le passage à l'étape suivante : l'ordonnancement des opérations. En vue d'accomplir l'équilibrage de la ligne et la commande en temps réel, on va tenir compte des paramètres et des équations présentés dans 3.2.2.2.

Tout d'abord il faut introduire les paramètres du processus : le nombre de postes et le nombre d'opérations de désassemblage. Puis l'utilisateur doit fournir le type de chaque opération, son nom et les temps opératoires. Deux types d'opérations de désassemblage sont pris en considération : les opérations de désassemblage propres et opérations de désassemblage destructives. Il y a des cas où une opération de désassemblage peut être soit propre soit destructive. Dans chaque cas les temps opératoires sont ajoutés comme données d'entrée. De même, la matrice d'affectation des opérations sur les postes de travail est introduite en utilisant le facteur $j$ ("Phy" sur l'interface) définit dans le paragraphe 3.2.2.2. Cette interface permet la commutation entre l'interface d'aide à la décision et l'interface de commande.

En vue de réaliser la commande du processus il reste encore à réaliser l'équilibrage de la ligne. La figure 41 présente l'interface nécessaire. Le paramètres importants sont : le temps restant de désassemblage, le nombre de produits restants à désassembler, le paramètre $y$ qui exprime l'état du processus de désassemblage (ou l'état d'opération), le type d'opération de désassemblage (propre ou destructive) et les temps de désassemblage. En fonction de l'état du processus, le paramètre $q$ est introduit par l'utilisateur. Puis, la valeur minimale de la fonction d'équilibrage donnée par l'équation (21) est calculée. Nous avons utilisé l'algorithme classique d'optimisation comme celui présenté dans le paragraphe 4.3.2. A la suite de ce calcul, la matrice d'affectation en temps réel des tâches est affichée. La commande des ressources est maintenant possible après l'action sur le bouton "Commande".

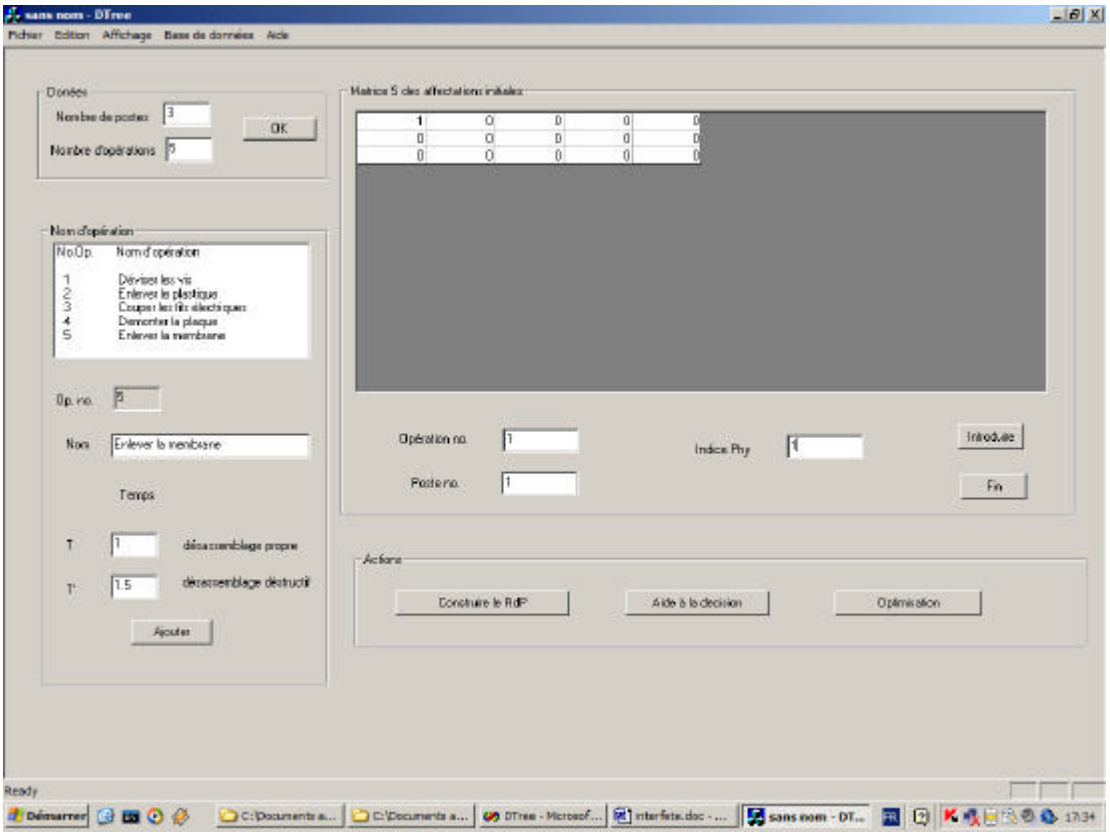



Fig. 41. L'interface de l'ordonnancement (première partie)

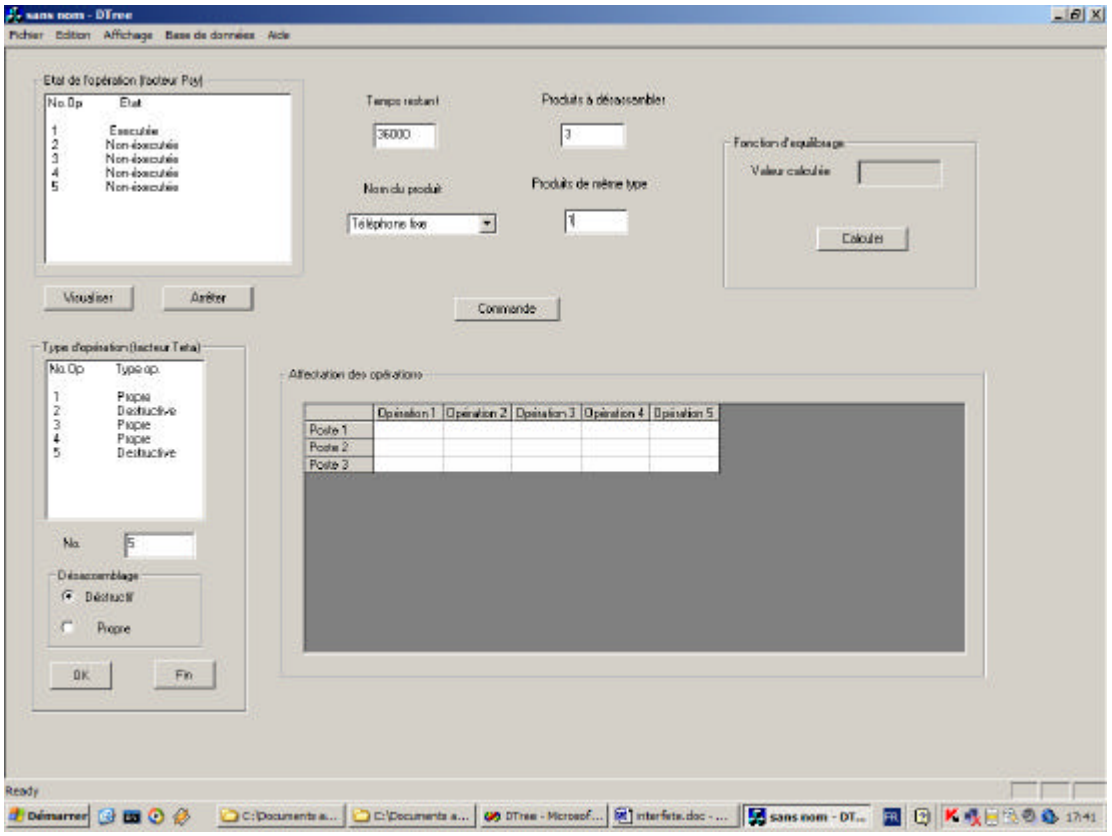



Fig. 42. L'interface de l'ordonnancement (deuxième partie)

Les menus d'application ont les mêmes fonctions que les boutons de l'interface. De plus, le menu "Base de données" permet l'accès rapide aux bases de données utilisées par l'application. En vue de familiariser l'utilisateur avec les notations utilisées sur l'interface, un guide sommaire est disponible dans le menu "Aide...".

### 5.4. Conclusions

Cette application est en fait une simulation de conduite d'un processus de désassemblage. En vue de perfectionner l'interface opérateur le logiciel doit être installé dans un système de désassemblage qui nécessite la commande en temps réel. Le déroulage du processus de désassemblage aidera à la conception d'une interface plus complexe que celle qui a été présentée. Une interface conçue sous Visual C++ est moins spectaculaire que celles conçues sous Visual Basic ou Delphi, mais les ressources offertes par ce logiciel sont plus complexes. De plus, tous les algorithmes d'optimisation ont été implémentés à l'aide du langage de programmation C++ de façon à avoir un travail unifié du point de vue informatique.

# CONCLUSION GÉNÉRALE

L'intérêt porté au désassemblage est lié aux notions de recyclage et de récupération des composants et de la matière des produits manufacturés en fin de vie. La plupart des recherches dans le domaine du désassemblage sont concentrées sur la planification du processus et sur la représentation des gammes de désassemblage. Face à au manque de travaux en commande en temps réel des systèmes de désassemblage nous avons proposé une solution visant l'optimisation du processus de désassemblage par une nouvelle approche de conduite des systèmes de désassemblage.

L'apport essentiel que représente l'ordonnancement statique et dynamique de la ligne de désassemblage s'est décliné en deux étapes. La première a été d'affecter les opérations à chaque poste de travail et d'établir ainsi l'ordre des tâches sur ces postes de manière à ce qu'il n'y ait pas de temps morts sur les différents postes. Il a été question d'ordonnancement hors ligne. La deuxième étape a consisté à maintenir l'équilibrage de la ligne et en temps réel optimisant notamment les ressources.

Il s'agit là d'un sujet évidement intéressant mais particulièrement difficile tant au niveau de l'incertitude qu'induit le processus de désassemblage des produits hors d'usage qu'à l'aspect du temps réel de l'ordonnancement de la ligne. Nous avons réussit à maîtriser ces difficultés par une approche considérant deux aspects : le problème d'optimisation par rapport au revenu final obtenu par le désassemblage du produit et le problème d'équilibrage de la ligne.

Le problème de la fonction objectif est un point délicat. C'est un problème multicritère avec des variables incertaines. Pour ce motif, nous avons proposé deux fonctions objectif : une fonction de coût et une fonction d'équilibrage. Le profil dynamique des deux problèmes et notre proposition d'une forme nouvelle de la fonction d'équilibrage a permis réellement d'effectuer l'évaluation des performances en temps réel de la ligne de désassemblage. De plus, les opérations destructives, dans le processus de désassemblage, ont été prises en considération.

En vue de l'optimisation des fonctions objectif, les méthodes classiques (backtracking) et celles approchées (algorithmes génétiques, algorithmes stochastiques) que nous avons utilisé et exploité ont permis d'avancer des conclusions utiles. La méthode classique et la méthode stochastique donnent des résultats similaires pour les problèmes de dimensions réduites. Les résultats différent pour plus des 30 taches et 10 stations de travail. L'algorithme Kangourou donne un optimum local; par contre un algorithme

combinatoire comme back tracking fournit un optimum global cherché parmis toutes les solutions possibles. Le point faible de l'algorithme classique est évidement la complexité exponentielle de la méthode qui augmente avec le nombre des données d'entrée. Dans le processus de désassemblage une solution rapide qui convienne au problème d'optimisation est souvent préférée.

La comparaison des résultats de simulations de ces méthodes a, en effet, mis en évidence qu'il était possible d'obtenir une affectation des tâches optimale à coût minimal. De plus, un bon équilibrage de la ligne de désassemblage a tout de même été obtenu. Une autre conclusion – évidente mais attendue – est que la méthode exacte fournit l'optimum global de la fonction objectif mais le désavantage principal est le développement exponentiel de la complexité de la méthode. Dans l'application des méthodes approchées, la progression vers l'optimum est facilitée par l'apport d'opérations stochastiques. Dans ce cas les algorithmes génétiques ou les algorithmes stochastiques deviennent avantageux du fait de la recherche effectuée dans une direction qui semble plus prometteuse. Les deux algorithmes donnent une bonne solution du problème dans un temps réduit par rapport à la méthode classique.

La dernière partie de la thèse à traité du problème de commande en temps réel. Un système d'aide à la décision est intégré dans l'architecture de commande. Les informations sont collectées en temps réel (événements, données, paramètres, décisions) et la commande est effectué à la suite d'analyse des ces informations. La prise de décision est faite à la suite de l'analyse de la valeur de la fonction objectif. En vue d'ameliorer l'expression de la fonction objectif en temps réel, un paramètre qui exprime l'état de chaque opération a été introduit. Ce paramètre est lu d'un ficher qui est actualisé en temps réel. Dans le même temps, le système d'aide à la décision reçoit des données en temps réel. Les informations sont utilisées pour décider quelle est l'opération suivante et à quel post elle est affféctée. De même, le type d'opération de désassemblage peut être choisi : opération déstructive ou propre. Entre le processus de désassemblage et le système de commande le changement d'information est permanente. Cet echangé est suivi à l'aide d'interface de contrôle et commande.

L'interface de commande (conçue en Visual C++ 6.0) a des fonctions pour réaliser l'ordonnancement en temps réel des produits sur la ligne et intègre une partie qui fait appel au système d'aide à la décision.

En perspective l'application pourra être testée sur un système de grand dimension, c'est à dire pour les produits complexes tels que des sous ensembles de voitures en fin de vie (tableau de bord, face avant,…). De même, au moment de l'implémentation de l'application sur un système de désassemblage réel, des corrections aux données collectées en temps réel seront apportées.

# RÉFÉRENCES


[Adamou, 1997] Adamou M. (1997) *Contribution à la modélisation en vue de la conduite des systèmes flexibles d'assemblage à l'aide des reseaux de Petri orientés objet*, Thèse de doctorat, Université de Franche Comté, France janvier 1997

[Addouche, 2003] Addouche S., (2003) *Contribution à une démarche de conception optimisée des processus de désassemblage*, Thèse de doctorat, Université de Franche-Comté, dec. 2003

[Addouche, 2002] Addouche S., (2002) Linear programming model to find the optimal disassembly sequence, *ICME, 2002, PORTUGAL*

[Agnetis, 1995] Agnetis A. (1995), Balancing flexible lines for a car components assembly, *International Journal of Production Research,* No. 33, pg 123-129

[Arai, 1993] Arai E., Iwata K. (1993), CAD system with product assembly/disassembly planning function, *Robotics&Computer-Integrated Manufacturing*, vol 10, no. 1-2, pg 41-48,1993

[Boothroyd, 1986] Boothroyd G., Dewhurst P., (1986) *Les assemblages, comment optimiser leur conception*, Manuel Assemblage, Paris, Edition CETIM, 1986

[Borne, 1992] Borne P, Dauphin-Tanguy G., (1992) *Modélisation et identification de processus*, Paris, Editions Techniup

[Bourjault, 1984] Bourjault A., (1984) *Contribution à une approche métodologique de l'assemblage automatisé : élaboration automatique des séquences opératoires*, Thèse d'Etat, Université de Franche Comté, 1984

[Bratcu, 2001] Bratcu A. (2001) *Détermination systématique des graphes de précédence et équilibrage des lignes d'assemblage*, Thèse de doctorat, Université de Franche-Compte,

Besançon, France, Juillet 2001

[Carlier, 1991] Carlier J., Chretienne P., (1991) *Les problèmes d'ordonnancement*, Rapport LAAS 91228, Grenoble,, Juin, 1991

[Chen, 1997] Chen S., Chou S., (1997), Parallel disassembly by onion peeling, *Journal of Mechanical Design*, vol 119, no. 2, pg. 267-274,

[Chevron, 1999] Chevron D. (1999), *Contribution à l'étude de la supervision d'une cellule de démontage de produits techniques en fin de vie*, Thèse de doctorat , Institut National Polytechnique de Grenoble

[Zaharie, 2001] Zaharie D., (2001) *Sisteme Informatice pentru Asistarea Deciziei*, (*Systèmes informatiques pour l'aide à la décision*) Bucarest, Editura DualTech, 2001

[David, 1999] David B., Saikali K., Bourtos N. (1999), Conception orientée recyclage des produits manufacturiers, *Le 3$^{ième}$ congrès International de Génie Industriel*, Montreal, Canada , mai 1999

[Delchambre, 1989] Delchambre A., Gaspart P., (1989), Knoledge-based assembly by dissaembly planning, *Proceedings of the International Conference on expert Systems in Engineering Applications*, China, 1989

[Dini, 2001] Dini G. Failli F., Santochi M., (2001), A disassembly planning software for the optimization of recycling processes, *Production Planning & Control*, Vol 12, No. 1, pg. 2-12, 2001

[DE, 1999] Directive Europeene, (1999) *Proposition de directive de Parlament Europeene dans ce qui concerne les réstrictions à la utilisation de substances toxiques dans les appareils électriques et élécroniques*, http://europa.eu.int/prelex/detail_dossier_real.cfm?CL=es&DosId=158021

[D0C, 2002] Document PSA (2000) *Rapport d'environnement* PSA 1999. http://www.psa.fr/enviro/environnement/fr/menu.html

[Dolgui, 2006] Dolgui Al., Guschinski N., Levin G., (2006) A special case of transfer line balancing by graph approach, *European Journal of Operational Research*, no. 168, 2006, pg. 732-746

[Duta, 2002a] Duta L., F. Filip, J. M. Henrioud (2002) – Conceptual Modeling for the Disassembly Processes, *Revue Studies in Informatics and Control*, June 2002, vol. 11, nr. 2, Pag. 177-184

[Duta, 2002b] Duta L, Filip Gh. F., (2002) Automatic Disassembly: Main Stage in Manufacturing Products, *Proceedings of the 4'th International Workshop on Computer Science and Information Technology*, September 2002, Patras, GREECE, on CD

[Duta, 2003a] Duta L, Filip, Gh. F., Henrioud J.M. (2003) An Algorithm for Dealing with Multi-Objective Optimisation Problem of Disassembly Processes, *Proceedings of the International Symposium of Assembly and Task Planning - ISATP 2003*, Pag. 163-168, July 2003, Besançon, France

[Duta, 2003b] Duta L, Filip, Gh. F., Henrioud J.M. (2003) Determination of the Optimal Disassembly Sequence Using Decision Trees, *Preprints The the International IFAC Workshop in Assembly and Disassembly IAD03*, 9-11 Oct. 2003,Pg. 49-55, Bucarest, Roumanie

[Duta, 2005] Duta L., Filip Gh. F, Henrioud J.M., (2005) Applying equal piles Approach to Disassembly line balancing problem, *Preprints of The 16th IFAC World Congress*, Prague, 4-8 July 2005, CD

[Dutta, 1995] Dutta D., Woo T., (1995), Algorithm for multiple disassembly and parallel assemblies, *Journal of Engineering for Industry*, vol. 117, no. 1, 1995

[Fernandez, 2001] Fernandez R., Zerhouni N., (2001), Modélisation et analyse des systèmes de désassemblage par réseaux de Petri continus, *Proceedings of the 2001 IEEE International Symposium on Assembly and Task Planning, Spain, 2001*

[Filip, 2005] Filip Gh. F. (2005) *Decizie asistata de calculator*, (*Systèmes informatiques de l'aide à la décision*) Bucarest, Ed. Tehnica, 2005

[Filip, 1999] Filip Gh. F., Barbat B., (1999) *Informatica Industriala - noi paradigme si aplicatii*, (*Informatique Industrielle*) Bucarest, Editura Tehnica, 1999

[Fleury, 1995] Fleury Gérard, (1995) Applications de méthodes stochastiquesninspirées du recuit simulé à des problèmes d'ordonnancement, *Informatique Industrielle*, Vol. 29, no 4-5, 1995

[Garbaya, 2003] Garbaya S., Coiffet Ph., Blazevic P., (2003) Experiments of Assembly ans Disassembly Planning in Virtual Environnement, Preprints of the International Symposium of Assembly and Task Planning, ISATP 2003 Besançon France, pg 37-43

[Gerner, 2001] Gerner S., (2001) *Génération d'un processus de désassemblage et évaluation du recyclage d'un produit*, Thèse de doctorat, Institut National Polytechnique de Grenoble, sept. 2001

[Gungor, 1999a] Gungor A., Gupta S. M., (1999), Disassembly line balancing, *Proceedings of Northeast Decision Sciences Institute*, Rhode-Island, march 1999

[Gungor, 1999b] Gungor A., Gupta S. (1999), A systematic solution approach to the dissasembly line balancing problem, *Proceedings of the 25th International Conference on*

*Computers and Industrial Engineering*, New Orleans, April, 1999, pp 70-73

[Gungor, 2001] Gungor A., Gupta S., (2001) A solution approach to the disassembly line balancing problem in the presence of task failures, *International Journal of Production and Research* , Vol. 39, No. 7, 2001, pp 1427-1467

[Gupta, 1994] Gupta S. M., Taleb K. N. (1994), Scheduling disassembly, *International Journal of Production and Research*, vol. 32, No. 8, pp. 1857-1866, 1994

[Gupta, 1996] Gupta S., Veerakamolmal P., (1996), *Disassembly of products*, Etats Unis, Final Raport, Grant no. 60NANB5D0112, 1996

[Heissel, 1995] Heissel H., Meyer-Krahmer K. (1995), Auf dem Weg zu einer okologischen Stoffwirtschaft. *Gaia – Ecological Perspectives in Science, Humanities, and Economics*, GAIA 4 no2, 1995

[Hennet, 1997] Hennet J. C., (1997) *Concepts et outils pour les systèmes de production*, France, Editions Cépaudes, 1997

[Henrioud, 1989] Henrioud J.M, (1989) *Contribution à la conceptualisation de l'assemblage automatisé : nouvelle approche en vue de la détermination des processus d'assemblage*, Thèse de doctorat, Université de Franche Comté, France, 1989

[Imtanavanich, 2004] Imtanavanich P, Gupta S., (2004) Multi-Criteria Decision Making for Disassembly-To-Order System Under Stochastic Yields, *Proceedings of SPIE The International Society for Optical Engineering*, pg 147-162. Vol.5583, Bellingham, WA, 2004

[Jackson, 1956] Jackson J. R., (1956) A computing procedure for a line balancing problem, *Management Science 2*, pg 261-271, 1956

[Johnson, 1998] Johnson M. R., Wang M. H, (1998) Economical evaluation of disassembly operations for recycling, remanufacturing and reuse, *International Journal of Production and research*, Vol 36, No 12, 3227-3252

[Johnson, 1981] Johnson R. V. (1981) Assembly line balancing algorithms: Computation comparisons, *International Journal of Production and Research*, no 19, pp. 277-287, 1981

[Kimmek, 1997] Kimmek M. (1997), *Fraktionierung technischer Gerbrauchsguter mittels eines frei beweglichen Wasserstrahlwerkzeugs*, Dissertation Universitat Karlsruhe, 1997

[Kirkke, 1998] Kirkke H., van Harten A., Schuur P. (1998), On a medium term product recovery and disposal strategy for durable assembly products, *International Journal of Production Research*, Vol. 36, no 1, 111-139, 1998

[Kizilkaya, 1998] Kizilkaya E., Gupta S. (1998), Material flow control and scheduling in a disassembly environment, *Computers Industrial Engineering*, vol. 35, no. 1, Pg. 93-96

[Kopacek, 2003] Kopacek P. and B. Kopacek, (2003) Robotized Disassembly of Mobile Phones, *Preprints of the International IFAC Workshop in Assembly and Disassembly IAD03*, 9-11 Oct. 2003,Bucarest,ROUMANIE, Pp 142-144

[Kopacek, 1998] Kopacek B, Kopacek P, (1998) Intelligent Disassembly of Electronic Equipement, *Proceedings of IFAC Assembly and Disassembly*, Bled, Slovenia, 1998, pp. 87-92

[Kopacek, 2005] Kopacek P. (2005) Semiautomized Disassembly, Some Examples, *Preprints of The 16th IFAC World Congress*, Prague, 4-8 July 2005, CD

[Kuo, 2000] Kuo T.C., Zhang H.C., Huang S.H. (2000), Disassembly analysis for electromechanical products: a graph based heuristic approach, *International Journal of*

*Production Research*, vol. 38, no.5, 2000

[Lambert, 2003] Lambert A.J.D. (2003) Disassembly sequencing: A survey, *International Journal of Production and Research*, vol 41, No 16, Pag 3721-3759, 2003

[Laperrière, 1992] Laperrière L., El Maraghy H. (1992), Planning of products assembly and disassembly, *Anals of the CIRP*, vol 41, pg 5-9, 1992

[Lee, 1992] Lee Y.Q., Kumana S. (1992), Individual and group disassembly sequence generation throught freedom and interference spaces, *Journal of Design and Manufacturing*, vol. 2, pg. 143-154, 1992

[Lopez, 2001] Lopez P., Roubellat F., (2001) *Ordonnancement de la production*, Paris, Ed. Hermes, 2001

[Mascle, 1990] Mascle C., Figour J., (1990) Methodological approach of sequences determination using the disassembly method, *Proceedings of the 2nd International Conference on Computer Integrates Manufacturing*, New-York, 1990, pp483-490

[McGovern, 2004] McGovern M., Gupta S, (2004) Combinatorial optimisation methods for disassembly line balancing, Proceedings of SPIE, *The International Society for Optical Engineering*, 5583, Pages 53-66, Bellingham, WA, 2004

[McGovern, 2003] McGovern M., Gupta S, Kamarthi S. V., (2003) Sloving disassembly sequence planning problems using combinatorial optimisation, *Proceedings of the 2003 Annual Meeting of the NorthEast Decision Sciences Institute*, March 2003, Rhode Island, pp. 178-180

[Moore, 1998] Moore K., Gungor A., Gupta S. (1998), Disassembly process planning using Petri Nets, *Proceedings of the IEEE International Symposium On Electronics and Environment*, Oak Brook, Illinois, USA, 1998

[Nof, 1997] Nof Sh., Wilhelm W., Warnecke H-J., (1997) *Industrial Assembly*, Chapman&Hall Edition, London, 1997

[Nollet, 1994] Nollet J., Kélada J., Diorio M., (1994) *La Gestion des Opérations et de la Production – Une approche systèmique*, Paris, Ed. Gaetan Morin, 1994

[Ontiveros, 2004] Ontiveros Miguel Angel Lopez, (2004) *Intégration des contraintes de remanufacturabilité en conception de produits*, Thèse de Doctorat, Instutut National Polytechnique de Grenoble, 2004

[O'Shea, 1998] O'Shea B, Kabernick H, Grewal, S., (1998) State of the art Literature Survey, *Disassembly Planning, Concurent Engineering: Research and Applications*, vol 6, No 4, Dec 1998

[Penev, 1996] Penev K.D., de Ron A. J., (1996**),** Determination of a disassembly strategy**,** *International Journal of Production and Research*, vol. 34, no. 2, 1996

[Perrard, 1996] Perrard C., Magerand E., Bourjault A., (1996), Extraction by disassembling of a single part of a mechanical system for its mainenance, *Proceedings of IEEE International Conference on Emerging Technologies and Factory Automation*, Hawaii, 1996

[Pham, 1999] Pham V. H., (1999) *Contribution à une méthodologie de conception des systèmes flexibles d'assemblage ou désassemblage*, Thèse de doctorat de l'Institut National des Sciences Appliquées de Lyon, 1999

[Puente, 2003] Puente S. T., Torres F., Aracil R., (2003) Non-destructive Disassembly Robot Cell for Demanufacturing Automation, *Preprints of the International IFAC Workshop in Assembly and Disassembly IAD03*, 9-11 Oct. 2003,Bucarest,ROUMANIE, Pg. 126-131

[Puente, 2002] Puente S.T. (2002) *Desensamblado automatico no destructivo para la reutilizacion de componentes. Aplication al desensamblado de PC's*, PhD Thesis at

the University of Alicante, Spain, 2002

[Qian, 1996] Qian W.H, Pagello E., (1996) On the scenario and heuristics of disassemblies, *Proceedings of IEEE International Conference on Emerging Technologies and Factory Automation*, Hawaii, 1996

[Rekiek, 2001] Rekiek B., (2001) *Assembly Line Design*, Ph.D. Thesis, Free University of Bruxelles, 2001.

[Rembold, 1994] Rembold U., Nnaji B.O., Storr A. (1994), *Computer Integrated Manufacturing and Engineering*, Addison –Wesley Publishing Company, 1994

[Salomonski, 1999] Salomonski N., Zussman E. (1999) On line predictive model for disassembly process planning adaptation, *Robotics and Computer Manufacturing* no 15 pg. 211-220, 1999

[Scholl, 2003] Scholl A., Becker C., (2003) State-of-the-art exact and heuristic solution procedures for simple assembly line balancing, *Jenaer Schriften zur Wirtschaftswissenschaft*, Friedrich-Schiller Universitat, Germany, no. 20/2003

[SNDD, 1999] Ministère de l'Environnement, Gouvernement du Roumanie, *(1999) Strategia Nationala pentru Dezvoltare Durabila*, Editura NOVA, 1999

[Spengles, 1998] Spengler Th. (1998), *Industrielles Stoffmanagement*, Berlin, Erich Schmidt Verlag, 1998

[Strege, 1995] Strege B., Weigl A., Gros A, (1995) Scheduling of disassembly cells based on Timed Extended Controlled Petri Nets, T*he 5th International Conference on Flexible Automation and Intelligent Manufacturing*, Sttutgart, Juin, 1995

[Subramani, 1991] Subramani A.K. , Dewhurst P. (1991), Automatic generation of product disassembly sequences, *Annals of the*

*CIRP*, vol 40, pg 115-118, 1991

[Taleb, 1998] Taleb K., Gupta, S. (1998) Disassembly of complex product structures with parts and materials commonality, *Production Planning&Control*, vol. 8, no.3, pg 255-269, 1998

[Torres, 2004] Torres, F., Gil, P., Puente, S.T., Pomares, J., Aracil, R. (2004) Automatic PC disassembly for component recovery *International Journal of Advanced Manufacturing Technology*, 23 (1-2), Pages 39-46.

[Touzanne, 2002] Touzanne F., (2002) *Contribution à une méthode de conception des systèmes de désassemblage des produits en fin de vie*, Thése de doctorat, Université de Franche Comté, Besançon, France, Juillet 2002

[Touzanne, 1999] Touzanne F., Perrard C. (1999), Representation of disassembly processes in order to allow time evaluation of their performances, *Proceedings of the 1999 IEEE International Symposium on Assembly and Task Planning*, Portugal, July,1999

[Wiendahl, 1999] Wiendahl H. P., Seliger G., Perlewitz H., Burkner S**,** (1999) A general approach to disassembly planning and control, *Production Planning and Control*, vol. 10. No 8, Pp 718-726, 1999

[Wiendahl, 2001] Wiendhal H. P., Brukner S., Lorenz B., (2001) The necessity of flexible and modular structures in hybrid disassembly processes, *Production Engineering*, No. 5, pg. 131-136

[Woo, 1991] Woo T., Dutta D. (1991) Automatic disassembly and total ordering in 3 dimensions, *ASME Journal of engineering for Industry*, pg 207-213, 1991

[Yin, 1995] Yin B. (1995) *Contribution au pilotage réactif des systèmes flexibles d'assemblage: méthode d'équilibrage dynamique*, Thèse de doctorat, Université de Franche-Comté, Besançon, France, Septembre 1995

[Zussman, 1999] Zussman E, Zhou M., (1999) Methodology for Modeling and Adaptive Planning of Disassembly processes, *IEEE Transactions on Robotics and Automation* vol. 15 no 1, 1999

# APPENDICE A

# LES CODES SOURCES DES ALGORITHMS UTILISÉS

**//Le Programme d'ordonnancement à l'aide d'une méthode classique (backtracking)**

```
#include <fstream.h>
#include <conio.h>
#include <math.h>
#define nrmaxop 30
#define nrmaxp 5
#typedef boolean {0,1}
short int mat_m[nrmaxp,nrmaxop] ;
short int mat_x[nrmaxop,nrmaxp];
short int stiva[nrmaxop];
short int  mat_m vec_s[1..nrmaxop];
 float mat_t[10,nrmaxop];
     float vec_q[2,nrmaxp] ;
     float vec_f[2,10];
stiva st,v ;
   mat_m  m,s;
    mat_x a;
    int nrp,nrop,n,i,j,k,nrsol,nrs,nrt,gama,mats;
    int x,y,z;
    boolean as,ev;
    vec_s vs;
    mat_t t;
     float tcy,min;
   vec_q  q,fq;
     vec_f f;

float calculf( mat_m si, int nrgama,  mat_t t)
{float q[nrmaxp] ; int i,j,k;float fq;
  for (k=1;k<=nrp; k++)
   { q[k]=0;
    for (j=1; j<=nrop ; j++)
      q[k]=q[k]+s[k][j]*t[nrgama][j]
   }
  fq=0;
  for (k=1;k<=nrp; k++)
   fq=fq+pow(q[k]-tcy);
  return fq;
}

void lire_t(float tcy, int nrt,mat_t t)
{ ifstream f("t.txt");int i;
 f>>tcy;
 nrt=0;
```

```
 while (!eof(f))
  {
   nrt=nrt+1;
   for (i=1;i<=nrop;i++)  f>>t[nrt][i];
   f>>eoln;
  }
 close(f);
}

void init(int k,stiva st)
{
 st[k]=0;
}

int succesor(int k,stiva st, boolean as)
{
 if (st[k]<v[k])
  {
   st[k]=st[k]+1;
   as=0;
  }
 else
  as=1;
return as;
}

int valid(int k,stiva st,boolean as)
{int i;
 ev=1;
 for (i=1; i<=k-1;i++)
  if (a[i+1][st[i+1]]<a[i][st[i]]) ev=0;}
return ev;
}

int solutie(int k)
{
return solutie=(k==n);
}

void tipar( )
{int i,j,k,s1,s2;fstream g;boolean da;
 nrsol=nrsol+1;
  for (i=1;i<=nrp;i++)
 for (j=1;j<=nrop;j++)  s[i][j]=0;
 for (i=1;i<=n;i++)
  s[ a[i][st[i]] [i]=1;

 da=true;
 for (i=1;i<=nrop;i++)
  s1=0;s2=0;
  for (j=i+1;j<=nrop;j++)
   {
  for (k=1;k<=nrp;k++)
    {s1=s1+k*s[k][i];
     s2=s2+k*s[k][j];
    }
  da=da && (s1-s2<=0);
  }
```

```
 }
 if da
 nrs=nrs+1;
 {  for (i=1;i<=nrp;i++)
for (j=1;j<=nrop;j++)  vs[nrs]s[i][j]=s[i][j];}
 vs[nrs]=s;
 ifstream g(''m.txt'');
 g<<'S'<<nrsol;
 for (i=1;i<=nrp;i++)
{  for (j=1;j<= nrop;j++) g<<s[i][j]<<' ';
  g<<eoln;
  }
 g<<eoln;
 close(g);
 }
}

void lire_m(int nrp,int nrop,mat_m m)
{ ifstream f(''m.txt'')
   int i,j;
 f>>nrp>>nrop;
for (i=1;i<=nrp;i++)
  for (j=1;j<=nrop;j++) f>>m[i][j];
 close(f);
}

void ecrire_m(int nrp,nrop;mat_m m)
{int i,j;

for (i=1;i<=nrp;i++)
  {
   for (j=1;j<=nrop;j++) cout<<m[i][j]<<' ';
    cout<<endl;
  }
 cout<<endl;
}

void ecrire_a(int nrop,int nrp, mat_x a)
{int i,j;
for (i=1;i<=nrop;i++)
   for (j=1;j<=nrp;j++) cout<<a[i][j]<<' ';
   cout<<endl;
  }
void main( )
{
cin>>nrp;
cin>>nrop;
 lire_m(nrp,nrop,m);
for (i=1;i<=nrop;i++)
 for (j=1;j<=nrp;j++) a[i][j]=0;

for (i=1;i<=nrop;i++)
{  k=0;
 for (j=1;j<=nrp;j++)
   if (m[j][i]==1)
{     k++;
     a[i][k]=j;}
  v[i]=k;
```

```
 }
 nrsol=0;nrs=0;
 n=nrop;
 k=1;
 init(k,st);
 while (k>0)
  { do
    {succesor(k,st,as);
    if as valid(k,st,ev);
   }while (as && ev);
   if as
    if solutie(k) tipar( )
    else
     {
      k++;
      init(k,st);
     }
   else
    k--;
 }

 for (k=1;k<=nrs;k++)
  {
  for (i=1;i<=nrp;i++)
      for (j=1;j<=nrop;j++) cout<<vs[[k][s[i][j]]<<' ';
   cout<<endl;
 }

 lire_t(tcy,nrt,t);
 {
 for (i=1;i<=nrt;i++)
{ for (j=1;j<=nrop;j++) cout<<t[i][j];
cout<<endl;
  }}

 for (i=1;i<=nrt;i++)
   for (k=1;k<=nrs;k++)
     {q[1][k]=calculf(vs[k],i,t);fq[2][k]=k;}
   min=fq[1][1];mats=ceil(fq[2][1]);
  for (k=1;k<=nrs;k++)
     if (min>fq[1][k] ) {min=fq[1][k];mats=ceil(fq[2][k]);}
   f[1][i]=min;f[2][i]=mats;
}
  min=f[1][1];gama=1;mats=ceil(f[2][1]);
  for (i=2;i<=nrt;i++)
   if (min>f[1][i] )
{min=f[1][i]; gama=i; mats=ceil(f[2][i]);}

  cout<<"Gamme optimale : ",gama," cout ", min);
  cout<<"Matrice S : ",mats);
  s=vs[mats];
   for (i=1;i<=nrp;i++)
{   for (j=1;j<=nrop;j++) cout<< s[i][j]<<' '); cout<<endl;
  }}
```

**// Le Programme de transformation du RdP dans ADD**

```
#include "stdafx.h"
#include<iostream.h>
#include<fstream.h>
#include<string.h>
struct Nod
{     int tip;
      float nv1,nv2,np1,np2;
      char chText[30];
} nod[50];

float matP[30][30];
int matPT[50][50];
int arrT[50],arrP[50];
int arrP1[50];

int main(int argc, char* argv[])
{
      int n,m,pi,pf,i,j,k,ok,nIndex,q;
      int coada[20],viz[20],t[20];
      float nv1,np1,nv2,np2;

      char filename[20],chText[20];
      ofstream fout;
      ifstream fin;
      cout<<"Filename=";
      cin>>filename;
    fin.open(filename);
      ok=1;
      nIndex=0;

      fin>>m>>n>>q;
      for(j=0;j<n;j++)
            arrP1[j]=0;
      for(i=0;i<m;i++)
      {

        for(j=0;j<n;j++)
            {
            fin>>matPT[i][j];
        if(matPT[i][j]==-1)
                  arrP1[j]=1;
            fin.ignore();
            cout<<matPT[i][j]<<" ";
            }
            cout<<endl;
      }

      t[nIndex]=-1;
      arrP[nIndex]=0;
      for(i=0;i<n;i++)
            arrP[i]=-1;
      for(j=0;j<m;j++)
          arrT[j]=-1;
    nIndex=0;
```

```
    for(i=0;i<n;i++)
      if(arrP1[i]!=0)
      {

            for(j=0;j<m;j++)
           if(matPT[j][i]!=0)
               {

                     if(arrP[i]==-1)
                     {
                           arrP[i]=nIndex;
                           cout<<"arrP["<<i<<"]="<<arrP[i]<<endl;
                           nod[nIndex].tip=1;
                           nIndex++;
                     }

                     if(arrT[j]==-1)
                           {
                           arrT[j]=nIndex;
                           cout<<"arrT["<<j<<"]="<<arrT[j]<<endl;
                           nod[nIndex].tip=0;
                           nIndex++;
                     }
                     if(matPT[j][i]==-1)
                       t[arrT[j]]=arrP[i];
                  else if (matPT[j][i]==1)
                      t[arrP[i]]=arrT[j];
               }
      }

       cout<<endl;

           for(i=0;i<n-q;i++)
       {
             fin>>j>>nv1>>chText;
             cout<<j<<" "<<nv1<<" "<<chText<<endl;
             nod[arrP[j-1]].nv1=nv1;
             nod[arrP[j-1]].np1=0;
             nod[arrP[j-1]].nv2=0;
             nod[arrP[j-1]].np2=0;
             strcpy(nod[arrP[j-1]].chText,chText);
       }
       for(i=0;i<nIndex;i++)
             for(j=0;j<nIndex;j++)
                   matP[i][j]=0;
       for(i=0;i<m;i++)
       {
             fin>>np1>>nv1>>np2>>nv2>>k>>chText;
             cout<<np1<<"   "<<nv1<<"   "<<np2<<"   "<<nv2<<"   "<<k<<"
"<<chText<<endl;
             strcpy(nod[arrT[i]].chText,chText);
             if(k!=0)
             {
                   matP[arrT[i]][arrP[k-1]]=np2;
                   nod[arrT[i]].np1=np1;
                   nod[arrT[i]].nv1=nv1;
                nod[arrT[i]].np2=0;
                nod[arrT[i]].nv2=0;
```

```
            }
            else
            {
                  nod[arrT[i]].np1=np1;
                  nod[arrT[i]].nv1=nv1;
                nod[arrT[i]].np2=np2;
                nod[arrT[i]].nv2=nv2;
            }

       }
            fin.close();

      for(i=0;i<nIndex;i++)
       {
            cout<<t[i]<<" ";
       }
      cout<<endl;
      for(i=0;i<nIndex;i++)
      {
          cout<<nod[i].chText<<" ";

      }

      cout<<endl;
      for(i=0;i<nIndex;i++)
      {
            for(j=0;j<nIndex;j++)
                  cout<<matP[i][j]<<" ";
            cout<<endl;
      }

      t[0]=-1;
      for(k=0;k<nIndex;k++)
      {
            viz[k]=0;
            coada[k]=-1;
      }
      int nr=nIndex;
    cout<<"Get File name:";
      cin>>filename;
      fout.open(filename);
      fout<<nr<<endl;

      fout<<0<<"  "<<1<<"  "<<1<<"  "<<nod[0].nv1<<"  "<<0<<"  "<<0<<"
"<<0<<" "<<0<<" "<<nod[0].chText<<endl;
      cout<<0<<"  "<<1<<"  "<<1<<"  "<<nod[0].nv1<<"  "<<0<<"  "<<0<<"
"<<0<<" "<<0<<" "<<nod[0].chText<<endl;

      coada[0]=0;
      viz[0]=1;
    ok=1;
      pi=0;
      pf=1;
      nIndex=1;

      while(ok==1)
      {
            ok=0;
```

```
         for(i=pi;i<pf;i++)
              for(j=0;j<nr;j++)
               if((viz[j]==0)&&(t[j]==coada[i]))
               {
                     ok=1;
//                   cout<<"j="<<j<<"     "<<"viz["<<j<<"]="<<viz[j]<<"
t["<<j<<"]="<<t[j]<<" i="<<i<<" coada["<<i<<"]="<<coada[i]<<endl;
                     viz[j]=1;
                     coada[nIndex]=j;
//cout<<t[j]+1<<"      "<<j+1<<"      "<<"      "<<nod[t[j]].chText<<"
"<<nod[j].chText<<endl;
cout<<t[j]+1<<"     "<<j+1<<"     "<<nod[j].tip<<"     "<<nod[j].nv1<<"
"<<nod[j].np1<<"  "<<nod[j].nv2<<"  "<<nod[j].np2<<"  "<<matP[i][j]<<"
"<<nod[j].chText<<endl;
fout<<t[j]+1<<"     "<<j+1<<"     "<<nod[j].tip<<"     "<<nod[j].nv1<<"
"<<nod[j].np1<<"  "<<nod[j].nv2<<"  "<<nod[j].np2<<"  "<<matP[i][j]<<"
"<<nod[j].chText<<endl;
                     nIndex++;
               }

               pi=pf;
               pf=nIndex;
      }
      fout.close();

      return 0;
}
```

**//Le programme de géneration de l'intérface des  figures 39 - 42**

```
// DTreeView.cpp : implementation of the CDTreeView class
//

#include "stdafx.h"
#include "DTree.h"
#include "DTreeDoc.h"
#include "DTreeView.h"
#include "ViewConstants.h"
#include "MainFrm.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////
///////
// CDTreeView

IMPLEMENT_DYNCREATE(CDTreeView, CFormView)

BEGIN_MESSAGE_MAP(CDTreeView, CFormView)
      //{{AFX_MSG_MAP(CDTreeView)
      ON_BN_CLICKED(IDC_ADAUGA, OnAdauga)
```

```
	ON_BN_CLICKED(IDC_RESET, OnReset)
	ON_BN_CLICKED(IDC_DECIZIE, OnDecizie)
	ON_BN_CLICKED(IDC_SANSA, OnSansa)
	ON_BN_CLICKED(IDC_ADAUGA2, OnAdauga2)
	ON_COMMAND(ID_FILE_OPEN, OnFileOpen)
	ON_BN_CLICKED(IDC_READFILE, OnReadfile)
	ON_WM_TIMER()
	ON_BN_CLICKED(IDC_START, OnStart)
	ON_BN_CLICKED(IDC_STOP, OnStop)
	ON_BN_CLICKED(IDC_PETRI, OnPetri)
	ON_COMMAND(ID_VIEW_ORDONANTARE, OnViewOrdonantare)
	ON_COMMAND(ID_VIEW_PETRI, OnViewPetri)
	ON_EN_CHANGE(IDC_EDIT1, OnChangeEdit1)
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////
///////
// CDTreeView construction/destruction

CDTreeView::CDTreeView()
	: CFormView(CDTreeView::IDD)
{
	//{{AFX_DATA_INIT(CDTreeView)
	m_strMess = _T("");
	//}}AFX_DATA_INIT
	// TODO: add construction code here

}

CDTreeView::~CDTreeView()
{
}

void CDTreeView::DoDataExchange(CDataExchange* pDX)
{
	CFormView::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CDTreeView)
	DDX_Control(pDX, IDC_TIME, m_time);
	DDX_Control(pDX, IDC_STOP, m_stop);
	DDX_Control(pDX, IDC_START, m_start);
	DDX_Control(pDX, IDC_NOD_NAME, m_node_name);
	DDX_Control(pDX, IDC_VENIT, m_venit);
	DDX_Control(pDX, IDC_SANSA, m_sansa);
	DDX_Control(pDX, IDC_RESET, m_reset);
	DDX_Control(pDX, IDC_TREE2, m_tree2);
	DDX_Control(pDX, IDC_TREE1, m_tree1);
	DDX_Control(pDX, IDC_TEXT, m_text);
	DDX_Control(pDX, IDC_ADAUGA2, m_adauga2);
	DDX_Control(pDX, IDC_ADAUGA, m_adauga);
	DDX_Control(pDX, IDC_NODES, m_nodes);
	DDX_Control(pDX, IDC_P12, m_p12);
	DDX_Control(pDX, IDC_PR, m_pr);
	DDX_Control(pDX, IDC_CA, m_ca);
	DDX_Control(pDX, IDC_PA, m_pa);
	DDX_Control(pDX, IDC_VA, m_va);
	DDX_Control(pDX, IDC_DECIZIE, m_decizie);
	DDX_Control(pDX, IDC_NOD2, m_nod2);
```

```
      DDX_Control(pDX, IDC_NOD1, m_nod1);
      DDX_Control(pDX, IDC_COMMONDIALOG1, dlg);
      DDX_Control(pDX, IDC_FG, m_grid);
      DDX_Text(pDX, IDC_EDIT1, m_strMess);
      //}}AFX_DATA_MAP
}

BOOL CDTreeView::PreCreateWindow(CREATESTRUCT& cs)
{
      // TODO: Modify the Window class or styles here by modifying
      //  the CREATESTRUCT cs

      return CFormView::PreCreateWindow(cs);
}

void CDTreeView::OnInitialUpdate()
{
      static_cast<CMainFrame*>(GetParentFrame())-
>SelectView(PETRI_FORM);
      GetParentFrame()->RecalcLayout();
      ResizeParentToFit();
    CFormView::OnInitialUpdate();
    OnDecizie();
      m_adauga2.EnableWindow(false);
      m_grid.SetFormatString("<Noeud   i  |<Noeud   j  |< Type  |< Perte
abandon   |<   Probabilité   d'abandoner   |<   Revenu   de   réussite   |<
Probabilité de réussite |< Probabilité |> Texte      ");
      OnReset();



}

/////////////////////////////////////////////////////////////////////////
///////
// CDTreeView diagnostics

#ifdef _DEBUG
void CDTreeView::AssertValid() const
{
      CFormView::AssertValid();
}

void CDTreeView::Dump(CDumpContext& dc) const
{
      CFormView::Dump(dc);
}

/*CDTreeDoc* CDTreeView::GetDocument() // non-debug version is inline
{
      ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CDTreeDoc)));
      return (CDTreeDoc*)m_pDocument;
}
*/
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////
///////
```

```
// CDTreeView message handlers

void CDTreeView::OnAdauga()
{

      // TODO: Add your control notification handler code here

      m_tree1.DeleteAllItems();
      m_tree2.DeleteAllItems();
      m_grid.SetRows(1);
      KillTimer(1);
      m_time.SetWindowText("");

      m_start.EnableWindow(true);
      m_stop.EnableWindow(false);
      CString szNodes;
      m_nodes.GetWindowText(szNodes);
      n=atoi(szNodes);
      if(n>0)
      {
            dlg.ShowSave();
        filename=dlg.GetFileName();

            m_start.EnableWindow(true);
            m_stop.EnableWindow(false);
            if (filename.GetLength())
            {
            fout.open(filename);
            fout<<n<<'\n';
            }
            m_grid.SetRows(n+1);
      }


      nPoz=0;
      m_adauga.EnableWindow(false);
      m_adauga2.EnableWindow(true);
      OnDecizie();
}

void CDTreeView::OnReset()
{
      // TODO: Add your control notification handler code here

      Reset();

}

void CDTreeView::OnDecizie()
{
      // TODO: Add your control notification handler code here
      //AfxMessageBox("
      m_decizie.SetCheck(1);
      m_sansa.SetCheck(0);
      m_pa.SetWindowText("");
      m_ca.SetWindowText("");
      m_pr.SetWindowText("");
      m_pa.EnableWindow(false);
```

```
	m_ca.EnableWindow(false);
	m_pr.EnableWindow(false);
	m_p12.EnableWindow(true);

	tip=1;

}

void CDTreeView::OnSansa()
{
	// TODO: Add your control notification handler code here
	tip=0;
	m_decizie.SetCheck(0);
	m_sansa.SetCheck(1);
	m_pa.EnableWindow(true);
	m_ca.EnableWindow(true);
	m_pr.EnableWindow(true);
	m_p12.EnableWindow(false);
	m_p12.SetWindowText("");

}

void CDTreeView::OnAdauga2()
{
	// TODO: Add your control notification handler code here
	CString szText;
	char chText[100];

	HTREEITEM h1;

	m_nod1.GetWindowText(szText);
	i=atoi(szText);
	m_nod2.GetWindowText(szText);
	j=atoi(szText);

	if(tip==1)
	{
		m_va.GetWindowText(szText);
		nv1=atof(szText);
		np1=0;
		np2=0;
		nv2=0;
		m_p12.GetWindowText(szText);
		np3=atof(szText);
		nPoz++;

	}
	else
	{
		m_va.GetWindowText(szText);
		nv1=atof(szText);
		m_pa.GetWindowText(szText);
		np1=atof(szText);
		m_ca.GetWindowText(szText);
		nv2=atof(szText);
		m_pr.GetWindowText(szText);
		np2=atof(szText);
		np3=0;
```

```
            nPoz++;
      }

          nod[j-1].tip=tip;
            nod[j-1].nv1=nv1;
            nod[j-1].np1=np1;
            nod[j-1].np2=np2;
            nod[j-1].nv2=nv2;
            m_text.GetWindowText(szText);
            wsprintf(chText,szText);
            if(filename.GetLength())
            {
          fout<<i<<" "<<j<<" "<<tip<<" "<<nv1<<" "<<np1<<" "<<nv2<<"
"<<np2<<" "<<np3<<" "<<chText<<endl;
            }
            nod[j-1].szName=szText;
            if(nod[j-1].tip==0)
            nod[j-1].venit=nv1*np1;
            else
             nod[j-1].venit=nv1;

            if(i!=0)//testare
            {
             mat[i-1][j-1]=1;
             mat[j-1][j-1]=1;
             matP[j-1][j-1]=np1;
             matP[i-1][j-1]=np3;
             t[j-1]=i-1;
           h[j-1]=m_tree1.InsertItem(nod[j-1].szName,h[i-1]);
             szText.Format("%.2f",nod[j-1].nv1);
             h1=m_tree1.InsertItem(szText,h[i-1]);
             //m_tree1.Expand(h[j-1],TVE_EXPAND);
              m_tree1.Expand(h1,TVE_EXPAND);

            }
            else
            {
             t[j-1]=i-1;
             h[0]=m_tree1.InsertItem(nod[0].szName);
             szText.Format("%.2f",nod[0].nv1);

             h1=m_tree1.InsertItem(szText);
             m_tree1.Expand(h[0],TVE_EXPAND);
             m_tree1.Expand(h1,TVE_EXPAND);

            }

                  if (np2!=0)
            {
                  nod[j-1].venit=nv1*np1+nv2*np2;
                  c[j-1]=1;
                  szText.Format("%.2f",nod[j-1].nv2);
                  h1=m_tree1.InsertItem(szText,h[i-1]);
                  m_tree1.Expand(h1,TVE_EXPAND);
                  //m_tree1.Expand(h[j-1],TVE_EXPAND);

            }
```

```
            m_grid.SetRow(nPoz);
            m_grid.SetCol(0);
            szText.Format("%d",i);
            m_grid.SetText(szText);
            m_grid.SetCol(1);
            szText.Format("%d",j);
            m_grid.SetText(szText);
            m_grid.SetCol(2);
            szText.Format("%d",tip);
            m_grid.SetText(szText);
            m_grid.SetCol(3);
            szText.Format("%.2f",nv1);
            m_grid.SetText(szText);
            m_grid.SetCol(4);
            szText.Format("%.2f",np1);
            m_grid.SetText(szText);
            m_grid.SetCol(5);
            szText.Format("%.2f",nv2);
            m_grid.SetText(szText);
            m_grid.SetCol(6);
            szText.Format("%.2f",np2);
            m_grid.SetText(szText);
            m_grid.SetCol(7);
            szText.Format("%.2f",np3);
            m_grid.SetText(szText);
            m_grid.SetCol(8);
            m_grid.SetText(nod[j-1].szName);

      if(nPoz==n)
      {
            AfxMessageBox("Gata");
            m_adauga2.EnableWindow(false);
            if (filename.GetLength())
            {
            fout.close();
            }

            //for(i=0;i<n-1;i++)
            //    fin<<i<<j<<tip<<nv1<<np1<<nv2<<np2<<np3<<chText;


      calcul();
      }

    m_nod1.SetWindowText("");
      m_nod2.SetWindowText("");
      m_va.SetWindowText("");
      m_pa.SetWindowText("");
      m_ca.SetWindowText("");
      m_pr.SetWindowText("");
      m_p12.SetWindowText("");
      m_text.SetWindowText("");

}


void CDTreeView::calcul()
{
```

```
    int ok=1;
      c[0]=1;
    int niv=1;
      while(ok==1)
      {
            ok=0;


            for(i=1;i<n;i++)
                  viz[i]=0;

             for(i=1;i<n;i++)
          if (c[i]==0)
                viz[t[i]]=1;

             for(i=1;i<n;i++)
             if ((c[i]==0)&&(viz[i]==0))
                   {

                         if (nod[i].tip==0)
                         {
                                 for(j=0;j<n;j++)
                                     if ((mat[i][j]==1)&&(i!=j))

nod[i].venit+=matP[i][j]*nod[j].venit;

                         }
                         else
                         {
                               max1=nod[i].venit;
                               for(j=0;j<n;j++)
                                     if
((mat[i][j]==1)&&(i!=j)&&(max1<nod[j].venit))

max1=nod[j].venit;

                               nod[i].venit=max1;

                         }

                         viz[i]=1;
                 c[i]=1;
                  }

             for(i=1;i<n;i++)
              if (c[i]==0)
              {
                    ok=1;
              }

      }
      max1=nod[0].venit;
      for(i=1;i<n;i++)
            if ((mat[0][i]==1)&&(max1<nod[i].venit))
                  max1=nod[i].venit;
    nod[0].venit=max1;
      CString szVal;
```

```
      i=0;
      int x;

  for(i=1;i<n;i++)
  {
      matV[i]=nod[i].venit;
       x=i;
       do
       {
      x=t[x];
      //    if((x!=-1)&&(nod[x].tip==0))
           if(x!=-1)
               matV[i]+=nod[x].venit;
       }
     while(x!=-1);

  }

matV[0]=nod[0].venit;
max1=matV[0];

int nodI;
nodI=0;

for(i=1;i<n;i++)
if (max1<matV[i])
{
      max1=matV[i];
      nodI=i;
}


x=nodI;
szVal.Format("%.2f",nod[x].venit);
m_venit.SetWindowText(szVal);
m_node_name.SetWindowText(nod[x].szName);
int nr1=0;
int tV[50];
tV[0]=x;
nr1++;
while(x!=-1)
{
      x=t[x];
            tV[nr1]=x;
            nr1++;

}
HTREEITEM h2[20],h3;


h2[0]=m_tree2.InsertItem(nod[0].szName);

szVal.Format("%.2f",nod[0].venit);
h3=m_tree2.InsertItem(szVal);

for(i=nr1-3;i>-1;i--)
{
```

```
      h2[tV[i]]=m_tree2.InsertItem(nod[tV[i]].szName,h2[tV[i+1]]);
    m_tree2.Expand(h2[tV[i+1]],TVE_EXPAND);
      szVal.Format("%.2f",nod[tV[i]].venit);
      h3=m_tree2.InsertItem(szVal,h2[tV[i+1]]);
      for(j=0;j<n;j++)
       if((tV[i+1]==t[j])&&(j!=tV[i]))
             h3=m_tree2.InsertItem(nod[j].szName,h2[t[j]]);

}

}

void CDTreeView::OnFileOpen()
{
      // TODO: Add your command handler code here
      int matPT[50][50];
    int arrT[50],arrP[50];
    int arrP1[50];
      char chText[50];
      CString szName;

      dlg.ShowOpen();
      filename=dlg.GetFileName();
      optFile=1;
      ReadData();

}
void CDTreeView::OnReadfile()
{
      m_adauga.EnableWindow(false);
      OnReset();
    dlg.ShowOpen();
      filename=dlg.GetFileName();
      optFile=0;
      m_start.EnableWindow(true);
      m_stop.EnableWindow(false);
      KillTimer(1);
      m_time.SetWindowText("");
      ReadData();
}

void CDTreeView::ReadData()
{
            // TODO: Add your control notification handler code here
      char chText[60];
      CString szText;
      HTREEITEM h1;
      m_tree1.DeleteAllItems();
      m_tree2.DeleteAllItems();
      optFile=1;
      ifstream fin(filename);
      m_grid.SetRows(1);
      fin>>n;
      m_grid.SetRows(n+1);
      m=n-1;
      for(i=0;i<n;i++)
      {
```

```
            for(j=0;j<n;j++)
          mat[i][j]=0;
            viz[i]=0;
            c[i]=0;
      }

      t[0]=-1;


    m_grid.SetColAlignment(0,2);
      for(k=0;k<m+1;k++)
      {
            m_grid.SetRow(k+1);
            fin>>i>>j>>tip>>nv1>>np1>>nv2>>np2>>np3>>chText;
            m_grid.SetCol(0);
            szText.Format("%d",i);
            m_grid.SetText(szText);
            m_grid.SetCol(1);
            szText.Format("%d",j);
            m_grid.SetText(szText);
            m_grid.SetCol(2);
            szText.Format("%d",tip);
            m_grid.SetText(szText);
            m_grid.SetCol(3);
            szText.Format("%.2f",nv1);
            m_grid.SetText(szText);
            m_grid.SetCol(4);
            szText.Format("%.2f",np1);
            m_grid.SetText(szText);
            m_grid.SetCol(5);
            szText.Format("%.2f",nv2);
            m_grid.SetText(szText);
            m_grid.SetCol(6);
            szText.Format("%.2f",np2);
            m_grid.SetText(szText);
            m_grid.SetCol(7);
            szText.Format("%.2f",np3);
            m_grid.SetText(szText);
            m_grid.SetCol(8);
            szText.Format("%s",chText);
          m_grid.SetText(szText);

            nod[j-1].tip=tip;
            nod[j-1].nv1=nv1;
            nod[j-1].np1=np1;
            nod[j-1].np2=np2;
            nod[j-1].nv2=nv2;
            nod[j-1].szName.Format("%s",chText);

            if(nod[j-1].tip==0)
            nod[j-1].venit=nv1*np1;
            else
             nod[j-1].venit=nv1;

            if(i!=0)
            {
             mat[i-1][j-1]=1;
             mat[j-1][j-1]=1;
```

```
            matP[j-1][j-1]=np1;
            matP[i-1][j-1]=np3;
            t[j-1]=i-1;

         h[j-1]=m_tree1.InsertItem(nod[j-1].szName,h[i-1]);
            szText.Format("%.2f",nod[j-1].nv1);
            h1=m_tree1.InsertItem(szText,h[i-1]);
            m_tree1.Expand(h[i-1],TVE_EXPAND);
           }
           else
           {
            t[j-1]=i-1;
            h[0]=m_tree1.InsertItem(nod[0].szName);
            szText.Format("%.2f",nod[0].nv1);
            h1=m_tree1.InsertItem(szText);
           }

           if (np2!=0)
           {
                 nod[j-1].venit=nv1*np1+nv2*np2;
                 c[j-1]=1;
                 szText.Format("%.2f",nod[j-1].nv2);
               h1=m_tree1.InsertItem(szText,h[i-1]);
           }

      }
      if (filename.GetLength())
      {
      fin.close();
      }
      calcul();

}

void CDTreeView::Reset()
{
      m_tree1.DeleteAllItems();
      m_tree1.DeleteAllItems();
      m_tree2.DeleteAllItems();
      m_grid.SetRows(1);
      m_venit.SetWindowText("");
      m_node_name.SetWindowText("");
      KillTimer(1);
      m_time.SetWindowText("");
      filename="";
      m_start.EnableWindow(true);
      m_stop.EnableWindow(false);
      n=0;m=0;
    tip=0;
    np1=0;nv1=0;
    nv2=0;np2=0;
    np3=0;
   for(i=0;i<120;i++)
   {
   t[i]=0;
   viz[i]=0;
   c[i]=0;
   matV[i]=0;
```

```
    for(j=0;j<120;j++)
    {
     mat[i][j]=0;
     matP[i][j]=0;
    }
   }
   m_adauga.EnableWindow(true);
   m_adauga2.EnableWindow(false);
   m_nodes.SetWindowText("");
   m_nod1.SetWindowText("");
   m_nod2.SetWindowText("");
   m_va.SetWindowText("");
   m_pa.SetWindowText("");
   m_ca.SetWindowText("");
   m_pr.SetWindowText("");
   m_p12.SetWindowText("");
   m_text.SetWindowText("");

}

void CDTreeView::OnTimer(UINT nIDEvent)
{
      // TODO: Add your message handler code here and/or call default
      CString szText;

      timp++;
    if(timp==10)
      {
      szText.Format("%d",timp);
      m_time.SetWindowText(szText);
    timp=0;
//    m_tree1.DeleteAllItems();
//    m_tree2.DeleteAllItems();

      ReadData();
//    calcul();
      }
      else
    {
      szText.Format("%d",timp);
      m_time.SetWindowText(szText);
      }


      CFormView::OnTimer(nIDEvent);
}

void CDTreeView::OnStart()
{
      // TODO: Add your control notification handler code here

      if (filename.GetLength())
      {
      timp=0;
      SetTimer(1,1000,NULL);
      m_start.EnableWindow(false);
      m_stop.EnableWindow(true);
      }
```

```
      else
      AfxMessageBox("Trebuie sa deschideti sau sa creati manual un
fisier in care sa fie definit arborele de decizie");

}

void CDTreeView::OnStop()
{
      // TODO: Add your control notification handler code here
      KillTimer(1);
      m_start.EnableWindow(true);
      m_stop.EnableWindow(false);
      timp=0;
      m_time.SetWindowText("");
//    OnReset();
}

void CDTreeView::OnPetri()
{
      // TODO: Add your control notification handler code here
      static_cast<CMainFrame*>(GetParentFrame())-
>SelectView(PETRI_FORM);

}

void CDTreeView::OnActivateView(BOOL bActivate, CView* pActivateView,
CView* pDeactiveView)
{
      // TODO: Add your specialized code here and/or call the base
class
      CDTreeDoc* pDoc=(CDTreeDoc*)GetDocument();
      if(pDoc->bReset==true)
            OnReset();
      else if(pDoc->filename2.GetLength()!=0)
      {

            optFile=1;
            filename=pDoc->filename2;
          ReadData();
            pDoc->filename2="";
      }


      CFormView::OnActivateView(bActivate,              pActivateView,
pDeactiveView);
}

void CDTreeView::OnViewOrdonantare()
{
      // TODO: Add your command handler code here
      static_cast<CMainFrame*>(GetParentFrame())-
>SelectView(ORDON_FORM);
}

void CDTreeView::OnViewPetri()
{
      // TODO: Add your command handler code here
```

```
      static_cast<CMainFrame*>(GetParentFrame())-
>SelectView(PETRI_FORM);

}

void CDTreeView::OnChangeEdit1()
{
      // TODO: If this is a RICHEDIT control, the control will not
      //   send   this   notification   unless   you   override   the
CFormView::OnInitDialog()
      // function and call CRichEditCtrl().SetEventMask()
      // with the ENM_CHANGE flag ORed into the mask.

      // TODO: Add your control notification handler code here

      m_strMess="désassemblage total";
      UpdateData(FALSE);}
```

**//Le programme d'application d'algorithme Kangourou**

```
// Ordon.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#include<math.h>
#include<iostream.h>
#include<fstream.h>
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include<conio.h>
#include<windows.h>
const maxnrop=12;
const maxnrp=12;
int u[maxnrp][maxnrop],v[maxnrp][maxnrop],us[maxnrp][maxnrop];
int l1[maxnrop],l2[maxnrop],l3[maxnrop];
int nrop,nrp,nrt,i,j,k;
int nr_prec;
int m[maxnrp][maxnrop],s[maxnrp][maxnrop];
float t[10][maxnrop];
float r[10];
float vprec[2][maxnrop],tprec[2][maxnrop],rprec[2][maxnrop];
float tcy;
float fs;

void citeste_prec()
{
      ifstream f;
      int k,ll1,ll2,ll3;

    f.open("prec.txt");
    k=0; i=0;

   while (!f.eof())
   {
```

```
        i++;
       f>>ll1>>ll3>>ll2;

        if(ll2==ll3)
              f>>tprec[0][i-1]>>tprec[1][i-1]>>rprec[0][i-
1]>>rprec[1][i-1];
        else
        {
              k++;
              l1[k-1]=ll1; l2[k-1]=ll2; l3[k-1]=ll3;
          f>>tprec[0][i-1]>>tprec[1][i-1]>>rprec[0][i-1]>>rprec[1][i-
1];
        }
   }

   f.close();
   nr_prec=k;

}

void citeste_t()
{
      ifstream f;
      int i;
      f.open("t.txt");
      f>>tcy;
      nrt=0;
      while(!f.eof())
      {

            for(i=0;i<nrop;i++)
            {
                  f>>t[nrt][i];
                  f.ignore();
        }
            f>>r[nrt];
       nrt++;
     }

      f.close();
}


void scrie_prec()
{
  int i;

  cout.precision(3);
  cout<<"Precedences :"<<endl;
  cout<<"L1 (Taches)    = ";


  for(i=0;i<nr_prec;i++)
        cout<<l1[i]<<" ";
  cout<<endl;

  cout<<"L2 (Taches)    = ";
  for(i=0;i<nr_prec;i++)
```

```
        cout<<l2[i]<<" ";
  cout<<endl;

  cout<<"L3 (Taches)    = ";
  for(i=0;i<nr_prec;i++)
        cout<<l3[i]<<" ";
  cout<<endl;

}
void scrie_mult(char c,int m[maxnrp][maxnrop])
{
      int i,j;
      cout<<"Une solution "<<c<<" :";
      for(i=0;i<nrp;i++)
      {
            cout<<"(";
            for(j=0;j<nrop;j++)
             if (m[i][j]==1)
                  cout<<j+1<<" ";
        cout<<") ";
      }
      cout<<endl;
      //getch();

}
int random(int n)
{
      Sleep(400);
      srand( (unsigned)time( NULL ) );

      return rand()%n;
}

float f(int m[maxnrp][maxnrop],int nrgama)
{
  float min,max,sum;
  int i,j;
  bool prim;

  min=0;

  for(j=0;j<nrop;j++)
        if (m[0][j]==1)
              min=min+vprec[0][j];

  for(i=1;i<nrp;i++)
      {
       sum=0;
       for(j=0;j<nrop;j++)
             if (m[i][j]==1)
                   sum=sum+vprec[0][j];
     if(sum<min)
             min=sum;
      }
  return min;

}
```

```
void gen_vecin(int u[][maxnrop],int v[][maxnrop])
{
      int x,oper,posti,postf,aux1;
      int i,j;
      float aux;

      x=random(nr_prec)+1;
      oper=l1[x-1];
      posti=l3[x-1];
      postf=l2[x-1];
      cout<<oper<<" "<<posti<<" "<<postf<<endl;

      for(i=0;i<maxnrp;i++)
            for(j=0;j<maxnrop;j++)
                  v[i][j]=u[i][j];

      v[postf-1][oper-1]=1;
      v[posti-1][oper-1]=0;

      aux1=l3[x-1]; l3[x-1]=l2[x-1]; l2[x-1]=aux1;
    aux=vprec[0][oper-1];            vprec[0][oper-1]=vprec[1][oper-1];
vprec[1][oper-1]=aux;

}

void descent(int u[][maxnrop],int us[][maxnrop],int& c)
{
      int v[maxnrp][maxnrop];
    int i,j;
      gen_vecin(u,v);
      cout<<"Voisinage generee par DESCENT"<<endl;
      scrie_mult('v',v);
      c=c+1;
      if (f(v,1)<=f(u,1))
      {
            if (f(v,1)<f(u,1))
            {
                  c=0;
                if(f(v,1)<f(u,1))
                  {
                        c=0;
                     if(f(v,1)<f(us,1))
                       {
                              for(i=0;i<maxnrp;i++)
                                    for(j=0;j<maxnrop;j++)
                                          us[i][j]=v[i][j];
                        fs=f(us,1);
                       }
                  }
             for(i=0;i<maxnrp;i++)
                        for(j=0;j<maxnrop;j++)
                              u[i][j]=v[i][j];
            }

      }
       cout<<"Solution optimale US de DESCENT "<<endl;
       scrie_mult('u',us);
       cout.precision(4); cout<<"fs="<<fs<<endl;
```

```
       cout.precision(2);

       for(i=0;i<nrop;i++)
              cout<<vprec[0][i]<<" ";
       cout<<endl;
       for(i=0;i<nrop;i++)
              cout<<vprec[1][i]<<" ";
       cout<<endl;
       getch();
}

void jump(int u[][maxnrop],int us[][maxnrop],int& c)
{
      int v[maxnrp][maxnrop];
    int i,j;
      gen_vecin(u,v);
      cout<<"Voisinage generee par JUMP"<<endl;
      scrie_mult('v',v);
      c=c+1;
    if(f(v,1)!=f(u,1))
      s{
            if(f(v,1)<f(us,1))
            {
                  for(i=0;i<maxnrop;i++)
                   for(j=0;j<maxnrop;j++)
                        us[i][j]=v[i][j];
                  fs=f(us,1);
            }
            c=0;
      }
      for(i=0;i<maxnrop;i++)
       for(j=0;j<maxnrop;j++)
         u[i][j]=v[i][j];
      cout<<"Solution optimale US de JUMP "<<endl;
      scrie_mult('u',us);
      cout.precision(4);
      cout<<"fs="<<fs<<endl;

      cout.precision(2);
      for(i=0;i<nrop;i++)
        cout<<vprec[0][i]<<" ";
      cout<<endl;
      for(i=0;i<nrop;i++)
        cout<<vprec[1][i]<<" ";
      cout<<endl;
      getch();

}

void kangourou()
{
      int a,c,i,j;
      a=4;
      c=1;
      for(i=0;i<nrp;i++)
            for(j=0;j<nrop;j++)
              us[i][j]=u[i][j];
    i=1;
```

```
      do
      {     if(c<a)
            {     descent(u,us,c);}
            else
                  jump(u,us,c);
            i++;
    }
      while(i<20);

}

void main( void )
{
    int i,j;

      nrp=3;
      nrop=8;

      for(i=0;i<maxnrp;i++)
            for(j=0;j<maxnrop;j++)
            u[i][j]=0;
      u[0][0]=1; u[0][1]=1; u[0][3]=1;
      u[1][2]=1; u[1][4]=1; u[1][5]=1;
      u[2][6]=1; u[2][7]=1;

      citeste_t();
      citeste_prec();
      for(i=0;i<nrop;i++)
      {  vprec[0][i]=tprec[0][i]/rprec[0][i];
         vprec[1][i]=tprec[1][i]/rprec[1][i];
      }

      cout<<"Liste de precedences "<<endl;
      scrie_prec();
      cout.precision(2);
      for(i=0;i<nrop;i++)
              cout<<vprec[0][i]<<" ";
      cout<<endl;
      for(i=0;i<nrop;i++)
        cout<<vprec[1][i]<<" ";
      cout<<endl;
      cout<<"START"<<endl;
      fs=f(u,1);
      cout.precision(4);
      cout<<"La premiere valeur de la f objectif fs= "<<fs<<endl;
    kangourou();
      scrie_mult('s',us);
      cout<<"fobj "<<fs<<endl;
      cout.precision(2);
      for(i=0;i<nrop;i++)
        cout<<vprec[0][i]<<" ";
      cout<<endl;
      for(i=0;i<nrop;i++)
        cout<<vprec[1][i]<<" ";
      cout<<endl;
      getch();}
```

# APPENDICE B

## LES FICHIERS D'ENTRÉE POUR LES PROGRAMMES SOURCES

**// Fichiers textes nécessaires pour les programmes d'optimisation**

**// Au désassemblage du téléphone :**

**T.txt**

**2.5**
**2 0 1.5 0 1.5 1.5**
**2 1.5 0 1 0 1.5**
**2 0 1.5 0 1 1**
**2 1 0 1 0 1**
**2 0 1.5 0 0 1**
**2 1.5 0 0 0 -2.5**
**2 0 0 0 0 -2**

**M.txt**

**2 5**
**1 1 0 0 1**
**0 1 1 1 1**

**// Au désassemblage de radio :**

**T.txt**

**1**
**0.12 0.07 0.07 0.12 0.95 0.75 0.75 0.95**
**0.15 0.07 0.07 0.12 0.95 0.75 0.75 0.95**
**0.15 0.07 0.07 0.15 0.95 0.75 0.75 0.95**
**0.15 0.07 0.07 0.12 0.95 0.9 0.75 0.95**

**M.txt**

**4 8**
**1 1 1 1 0 0 0 0**
**1 0 0 1 1 0 0 0**
**0 0 0 0 0 1 1 0**
**0 0 0 0 0 0 1 1**

**// Pour l'algorithme Kangoroo, le fichier Prec.txt donne la liste des précedences :**

**Prec.txt**

**1 1 2 0.12 0.15 -1.36 -1.32**
**2 1 1 0.07 0.07 0.27 0.27**
**3 2 2 0.07 0.07 0.54 0.54**
**4 1 2 0.12 0.15 0.54 0.5**
**5 2 2 0.95 0.95 0.62 0.62**
**6 2 3 0.75 0.90 0.67 0.60**
**7 3 3 0.75 0.75 2.75 2.75**
**8 3 3 0.95 0.95 2.75 2.75**

**T.txt**
**1**
**2 1 2 1 2 1.5 2 2 3 1 1.5**

## RÉSUMÉ

L'intérêt porté au désassemblage est lié aux notions de recyclage et de récupération des composants et de la matière des produits manufacturés en fin de vie. La plupart des recherches dans le domaine du désassemblage sont concentrées sur la planification du processus et sur la représentation des gammes de désassemblage. Face au manque de travaux en commande en temps réel des systèmes de désassemblage nous avons proposé une solution visant l'optimisation du processus de désassemblage par une nouvelle approche de conduite des systèmes de désassemblage. La définition des équations de l'ordonnancement du désassemblage en temps réel garantie le réalisme du problème et nous assure l'originalité de l'approche. Une architecture de système de commande est proposée. Les interfaces de commande sont conçues sous l'environnement Visual C++ 6.0. Elles intègrent une analyse de l'équilibrage de la ligne et un module d'aide à la décision.



## SUMMARY

The interest on the disassembly process is directly linked on the product recycling and recovering at the end of its life. Most of the researches in the disassembly field are concetrated on the level of process planning and disassembly sequencing. The lack of studies in the field of control of the disassembly process made us to propose a new solution regarding the process optimisation using an original approach. The definition of the real time sequencing of the disassembly process ensures the realism of the problem and gives an original feature to our work. An architecture for controling the disassembly process is proposed. Interfaces are entirely conceived under Visual C++ and they integrate modules of decision support.